

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : **06-243113**

(43)Date of publication of application : **02.09.1994**

(51)Int. Cl.

**G06F 15/16**

(21)Application number : **05-030971**

(71)Applicant : **FUJITSU LTD**

(22)Date of filing : **19.02.1993**

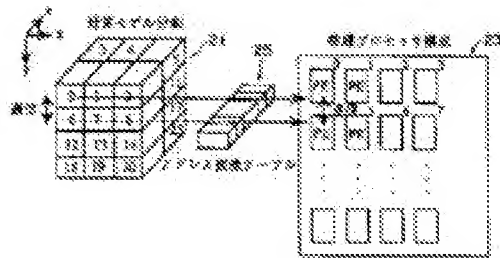
(72)Inventor : **SHIBATA KAZUYA  
FUJISAKI MASAHIRO  
KANAZAWA HIROYUKI  
OKUDA MOTOI**

## (54) CALCULATION MODEL MAPPING METHOD FOR PARALLEL COMPUTER

(57)Abstract:

PURPOSE: To develop and operate the applications with no consciousness of the architecture of a parallel computer by securing the correspondence between the calculation units divided from a calculation model and the physical processors via an address conversion table.

CONSTITUTION: An N-dimensional calculation model 21 is divided into plural calculation units having the identification codes. Meanwhile a physical processor constitution 23 has a two-dimensional array. The identification codes are given to plural physical processors PE. The correspondence is secured between the calculation units and the processors PE by the identification codes and via an address conversion table 25. Under such conditions, those divided calculation units are optionally mapped to the processors PE via a user interface. Then a user can perform the communication among those



processors by means of the identification codes of calculation units. Thus the work is attained in a user space and the mapping can be freely operated in accordance with the model 21.

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平6-243113

(43)公開日 平成 6 年(1994) 9 月 2 日

(51)Int.Cl.<sup>5</sup>

G 0 6 F 15/16

識別記号

庁内整理番号

3 9 0 T 9190-5L

F I

技術表示箇所

審査請求 未請求 請求項の数31 O L (全 28 頁)

(21)出願番号 特願平5-30971

(22)出願日 平成 5 年(1993) 2 月19日

特許法第30条第 1 項適用申請有り 1992年 8 月20日 社  
団法人情報処理学会発行の「情報処理学会研究報告92-  
N A-42」に発表

(71)出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中1015番地

(72)発明者 柴田 一哉

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72)発明者 藤崎 正英

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(72)発明者 金澤 宏幸

神奈川県川崎市中原区上小田中1015番地

富士通株式会社内

(74)代理人 弁理士 遠山 勉 (外 1 名)

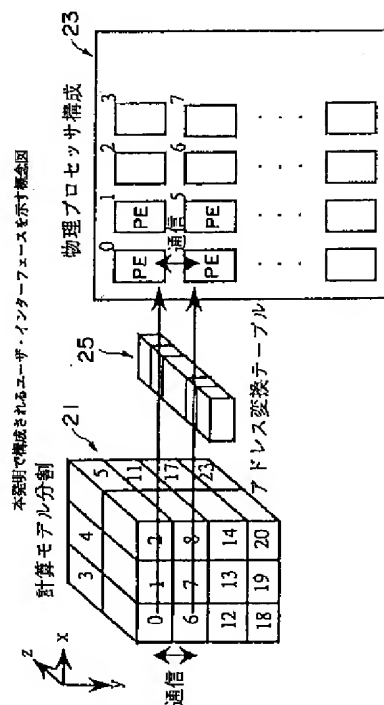
最終頁に続く

(54)【発明の名称】 並列計算機における計算モデルのマッピング法

(57)【要約】

【目的】 ユーザ空間で並列計算機のアーキテクチャを  
意識することなくマッピングができ、かつ、高速なマッ  
ピングパターンを得ることを目的とする。

【構成】 ユーザが分割したN次元計算モデル21の計  
算ユニットと、物理プロセッサ23に識別符号をつけて  
おき、その対応をアドレス変換テーブル25でもつこれ  
にユーザがアクセスするだけで、自由にマッピング操作  
ができるので、その後は、プロセッサ間通信はユーザが  
分割ユニットの識別符号を用いて行うことが出来る。



## 【特許請求の範囲】

【請求項1】 複数のプロセッサの間で相互にデータ及び情報を通信手段により転送することで各々のプロセッサが並列に処理を実行する並列計算機において、ユーザが演算処理しようとするN次元計算モデルを、前記複数のプロセッサで並列処理する方法であり、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割するステップと、前記分割した計算ユニットと各プロセッサとを対応づけるステップと、でユーザ・インターフェースを形成し、このユーザ・インターフェースを介して、分割した計算ユニットを、各プロセッサに任意にマッピングするステップを有し、その後、プロセッサ間通信を分割ユニットの識別符号を用いて行うことを特徴とする並列計算機における計算モデルのマッピング法。

【請求項2】 請求項1において、通信コストの評価関数として、プロセッサ間の距離の総和Lを用い、前記マッピングは、この距離の総和Lが最小となるように行うことを特徴とする並列計算機における計算モデルのマッピング法。

【請求項3】 請求項2において、分割ユニットの隣接格子点 $P_i$ 、 $P_j$ が、それぞれ、2次元格子点上のある点 $(X_i, Y_i)$ 、 $(X_j, Y_j)$ にマッピングされるとき、 $P_i$ 、 $P_j$ 間の距離 $l_{ij}$ は、 $|X_i - X_j| + |Y_i - Y_j|$ と表され、前記評価関数 $L = \sum l_{ij}$ とし、これが最小となったときのマッピングを最適なものと採用することを特徴とする並列計算機における計算モデルのマッピング法。

【請求項4】 請求項2または3において、初期マップ位置を決定した後、マップ位置を変化させ、長さの変化（前記距離の総和の変化） $\Delta L$ を算出し、 $\Delta L < 0$ のとき、前記マップ位置の変更を採用し、 $\Delta L < 0$ でないとき、乱数 $X$  ( $0 < X < 1$ )を引き出して $\exp(-\Delta L/T)$ を演算し（ $T$ は確立を決める目安となるパラメータ）、 $\exp(-\Delta L/T) < X$ のとき、マップ位置の変更を採用せず、 $\exp(-\Delta L/T) < X$ でないとき、マップ位置の変更を採用することとし、以上からの処理を、前記 $T$ の値を減少させて繰り返すことを特徴とする並列計算機における計算モデルのマッピング法。

【請求項5】 請求項1において、前記マッピングは、3次元計算モデルを、XYZ方向の3方向の計算ユニットに立体的に分割し、それらを2次元に配列されたプロセッサにマッピングする場合であり、前記3次元計算モデルをZX平面で $n$ 個の計算ユニットに分割しておき、前記2次元において、Y方向の各計算ユニットが正方形で隣合うようにし、ZX方向の計算ユ

ニットを一つ飛びに並べることを特徴とする並列計算機における計算モデルのマッピング法。

【請求項6】 請求項5において、(a) 3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義するとともに、マッピング対象となる2次元物理プロセッサにおいて、X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化することを前提とし、(b) 3次元計算モデルを $VX = PCX/2$ 、 $VY = PCY/2$ 、 $VZ = 0$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号（メインユニット番号）を付与し、さらにこのメインユニットをZ軸方向に4つに分割し、この結果できたユニットをサブユニットと呼んで、各分割単位に通し番号（サブユニット番号）を付与し、2次元物理プロセッサにおいて、 $PX = PCX/2$ 、 $PY = PCY/2$ に分割してプロセッサをグループ化し、各分割単位に通し番号（グループ番号）を付与し、(c) 3次元モデルのメインユニット番号と同一番号のグループ番号を有する物理プロセッサのグループにサブユニット番号0から3を、0と1、1と2、2と3、3と0とが隣接するように分配してマッピングすることを特徴とする並列計算機における計算モデルのマッピング法。

【請求項7】 請求項1において、前記マッピングは、3次元計算モデルを、XYZ方向の3方向の計算ユニットに立体的に分割し、それらを2次元に配列されたプロセッサにマッピングする場合であり、前記3次元計算モデルをZX平面で $n$ 個の計算ユニットに分割しておき、前記2次元において、各計算ユニットを順番に置き、ZX方向が隣合いY方向が均等（ $n$ 個分離れる）に並べたことを特徴とする並列計算機における計算モデルのマッピング法。

【請求項8】 請求項7において、(a) 3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義するとともに、マッピング対象となる2次元物理プロセッサにおいて、X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化することを前提とし、(b) 3次元計算モデルを $VX = 0$ 、 $VY = 0$ 、 $VZ = VCZ$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号（メインユニット番号）を付与し、このメインユニットのX方向計算単位数、Y方向計算単位数を $v_x$ 、 $v_y$ と

し、

2次元物理プロセッサにおいて、 $PX = PCX / vx$ 、 $PY = PCY / vy$ に分割してプロセッサをグループ化し、各分割単位に通し番号（グループ番号）を付与し、（c） $vx = PX$ 、 $vy = PY$ の位置に対応して、メインユニットを2次元物理プロセッサのX方向に通し番号順に配置し、そのライン上でのX方向の端部物理プロセッサに来たら、Y方向へ移行し、今度はメインユニットを逆のX方向に通し番号順に配置し、再度X方向の端部物理プロセッサに来たら、Y方向へ移行し、以後、以上の蛇行によるマッピングを繰り返すことを特徴とする並列計算機における計算モデルのマッピング法。

【請求項9】 請求項4の処理を並列処理することとを特徴とする並列計算機における計算モデルのマッピング法。

【請求項10】 請求項1において前記マッピングで得られた最適マッピング・パターンをデータベースに蓄積し、以後、このデータベースを参照して、計算モデルに適合する最適マッピングを選択し、並列処理をすることを特徴とする並列計算機における計算モデルのマッピング法。

【請求項11】 複数のプロセッサの間で相互にデータ及び情報を通信手段により転送することで各々のプロセッサが並列に処理を実行する並列計算機において、ユーザが演算処理しようとするN次元計算モデルを、前記複数のプロセッサで並列処理するにあたり使用する通信ライブラリであり、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割する分割ルーチンと、分割した計算ユニットの識別符号と前記各プロセッサに付与された識別符号との対応関係をアドレス変換テーブル上に形成する管理ルーチンと、分割した計算ユニットを、各プロセッサに任意にマッピングするマッピングルーチンと、を備え、プロセッサ間通信を分割ユニットの識別符号を用いて行うことを特徴とする並列計算機用通信ライブラリ。

【請求項12】 請求項11において、前記マッピングルーチンにおける通信コストの評価関数として、プロセッサ間の距離の総和Lを用い、前記マッピングは、この距離の総和Lが最小となるように行うことを特徴とする並列計算機用通信ライブラリ。

【請求項13】 請求項12において、分割ユニットの隣接格子点 $P_i$ 、 $P_j$ が、それぞれ、2次元格子点上のある点 $(X_i, Y_i)$ 、 $(X_j, Y_j)$ にマッピングされるとき、

$P_i$ 、 $P_j$ 間の距離 $l_{ij}$ は、 $|X_i - X_j| + |Y_i - Y_j|$ と表され、

前記評価関数 $L = \sum l_{ij}$ とし、これが最小となったときのマッピングを最適なものとして採用することを特徴

とする並列計算機用通信ライブラリ。

【請求項14】 請求項12または13において、初期マップ位置を決定した後、マップ位置を変化させ、長さの変化（前記距離の総和の変化） $\Delta L$ を算出し、 $\Delta L < 0$ のとき、前記マップ位置の変更を採用し、 $\Delta L < 0$ でないとき、乱数 $X$ （ $0 < X < 1$ ）を引き出して $\exp(-\Delta L/T)$ を演算し（ $T$ は確立を決める目安となるパラメータ）、 $\exp(-\Delta L/T) < X$ のとき、マップ位置の変更を採用せず、 $\exp(-\Delta L/T) < X$ でないとき、マップ位置の変更を採用することとし、以上からの処理を、前記 $T$ の値を減少させて繰り返すことを特徴とする並列計算機用通信ライブラリ。

【請求項15】 請求項11において、前記マッピングは、3次元計算モデルを、XYZ方向の3方向の計算ユニットに立体的に分割し、それらを2次元に配列されたプロセッサにマッピングする場合であり、前記3次元計算モデルをZX平面で $n$ 個の計算ユニットに分割しておき、前記2次元において、Y方向の各計算ユニットが正方形で隣合うようにし、ZX方向の計算ユニットを一つ飛びに並べることを特徴とする並列計算機用通信ライブラリ。

【請求項16】 請求項15において、（a）3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位を $VCX$ 、 $VCY$ 、 $VCZ$ と定義し、X方向、Y方向、Z方向の分割数を $VX$ 、 $VY$ 、 $VZ$ と定義するとともに、

マッピング対象となる2次元物理プロセッサにおいて、X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれ $PCX$ 、 $PCY$ と定義するとともに、X方向、Y方向に分割した分割数をそれぞれ $PX$ 、 $PY$ と定義して物理プロセッサをグループ化することを前提とし、（b）3次元計算モデルを $VX = PCX / 2$ 、 $VY = PCY / 2$ 、 $VZ = 0$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号（メインユニット番号）を付与し、さらにこのメインユニットをZ軸方向に4つに分割し、この結果できたユニットをサブユニットと呼んで、各分割単位に通し番号（サブユニット番号）を付与し、

2次元物理プロセッサにおいて、 $PX = PCX / 2$ 、 $PY = PCY / 2$ に分割してプロセッサをグループ化し、各分割単位に通し番号（グループ番号）を付与し、

（c）3次元モデルのメインユニット番号と同一番号のグループ番号を有する物理プロセッサのグループにサブユニット番号0から3を、0と1、1と2、2と3、3と0とが隣接するように分配してマッピングすることを特徴とする並列計算機用通信ライブラリ。

【請求項17】 請求項11において、前記マッピングは、3次元計算モデルを、XYZ方向の3方向の計算ユニットに立体的に分割し、それらを2次元に配列された

プロセッサにマッピングする場合であり、前記3次元計算モデルをZX平面でn個の計算ユニットに分割しておき、前記2次元において、各計算ユニットを順番に置き、ZX方向が隣合いY方向が均等(n個分離れる)に並べたことを特徴とする並列計算機用通信ライブラリ。

【請求項18】 請求項17において、(a)3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義するとともに、

マッピング対象となる2次元物理プロセッサにおいて、X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化することを前提とし、(b)3次元計算モデルを $VX=0$ 、 $VY=0$ 、 $VZ=VCZ$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号(メインユニット番号)を付与し、このメインユニットのX方向計算単位数、Y方向計算単位数を $v_x$ 、 $v_y$ とし、

2次元物理プロセッサにおいて、 $PX=PCX/v_x$ 、 $PY=PCY/v_y$ に分割してプロセッサをグループ化し、各分割単位に通し番号(グループ番号)を付与し、(c) $v_x=PX$ 、 $v_y=PY$ の位置に対応して、メインユニットを2次元物理プロセッサのX方向に通し番号順に配置し、そのライン上でのX方向の端部物理プロセッサに来たら、Y方向へ移行し、今度はメインユニットを逆のX方向に通し番号順に配置し、再度X方向の端部物理プロセッサに来たら、Y方向へ移行し、以後、以上の蛇行によるマッピングを繰り返すことを特徴とする並列計算機用通信ライブラリ。

【請求項19】 請求項14の処理を並列処理することを特徴とする並列計算機用通信ライブラリ。

【請求項20】 請求項11において前記マッピングで得られた最適マッピング・パターンをデータベースに蓄積するルーチンを有し、以後、このデータベースを参照して、計算モデルに適合する最適マッピングを選択し、並列処理をすることを特徴とする並列計算機用通信ライブラリ。

【請求項21】 複数のプロセッサの間で相互にデータ及び情報を通信手段により転送することで、ユーザが演算処理しようとするN次元計算モデルを、前記複数のプロセッサで並列処理する並列計算機において、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割する計算モデル分割部と、前記複数の各プロセッサに識別符号をつけておき、分割した計算ユニットと各プロセッサとの対応関係をアドレス変換テーブル上に形成する管理部と、

分割した計算ユニットを、各プロセッサに任意にマッピングするマッピング部と、を備え、プロセッサ間通信を分割ユニットの識別符号を用いて行うことを特徴とする並列計算機。

【請求項22】 請求項21において、前記マッピング部は、通信コストの評価関数として、プロセッサ間の距離の総和Lを演算するプロセッサ間距離演算部と、距離の総和Lが最小となったときのマッピングを最適なものとして採用することを特徴とする並列計算機。

【請求項23】 請求項22において、前記マッピング部は、前記プロセッサ間距離演算部で得られた距離の総和Lが最小であるか否かを判定する判定部とを有し、距離の総和Lが最小となったと判定部が判定したときのマッピングを最適なものとして採用することを特徴とする並列計算機。

【請求項24】 請求項21において、分割ユニットの隣接格子点Pi、Pjが、それぞれ、2次元格子点上のある点(Xi、Yi)(Xj、Yj)にマッピングされるとき、

Pi、Pj間の距離 $l_{ij}$ は、 $|X_i - X_j| + |Y_i - Y_j|$ と表され、

前記評価関数 $L = \sum l_{ij}$ とし、このLが最小であるとされたときのマッピングを最適なものとして採用することを特徴とする並列計算機。

【請求項25】 請求項22または23において、初期マップ位置を決定した後、マップ位置を変化させ、長さの変化(前記距離の総和の変化) $\Delta L$ を算出し、 $\Delta L < 0$ のとき、前記マップ位置の変更を採用し、 $\Delta L < 0$ でないとき、乱数X( $0 < X < 1$ )を引き出して $\exp(-\Delta L/T)$ を演算し(Tは確立を決める目安となるパラメータ)、 $\exp(-\Delta L/T) < X$ のとき、マップ位置の変更を採用せず、 $\exp(-\Delta L/T) < X$ でないとき、マップ位置の変更を採用することとし、以上からの処理を、前記Tの値を減少させて繰り返すことを特徴とする並列計算機。

【請求項26】 請求項21において、前記マッピング部は、3次元計算モデルを、XYZ方向の3方向の計算ユニットに立体的に分割し、それらを2次元に配列されたプロセッサにマッピングする場合であり、前記3次元計算モデルをZX平面でn個の計算ユニットに分割しておき、前記2次元において、Y方向の各計算ユニットが正方形で隣合うようにし、ZX方向の計算ユニットを一つ飛びに並べることを特徴とする並列計算機。

【請求項27】 請求項26において、(a)3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義するとともに、マッピング対象となる2次元物理プロセッサにおいて、

X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化することを前提とし、(b) 3次元計算モデルを $VX=PCX/2$ 、 $VY=PCY/2$ 、 $VZ=0$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号(メインユニット番号)を付与し、さらにこのメインユニットをZ軸方向に4つに分割し、この結果できたユニットをサブユニットと呼んで、各分割単位に通し番号(サブユニット番号)を付与し、

2次元物理プロセッサにおいて、 $PX=PCX/2$ 、 $PY=PCY/2$ に分割してプロセッサをグループ化し、各分割単位に通し番号(グループ番号)を付与し、

(c) 3次元モデルのメインユニット番号と同一番号のグループ番号を有する物理プロセッサのグループにサブユニット番号0から3を、0と1、1と2、2と3、3と0とが隣接するように分配してマッピングすることを特徴とする並列計算機。

【請求項28】 請求項21において、前記マッピング部は、3次元計算モデルを、XYZ方向の3方向の計算ユニットに立体的に分割し、それらを2次元に配列されたプロセッサにマッピングする場合であり、

前記3次元計算モデルをZX平面でn個の計算ユニットに分割しておき、前記2次元において、各計算ユニットを順番に置き、ZX方向が隣合いY方向が均等(n個分離れる)に並べたことを特徴とする並列計算機。

【請求項29】 請求項28において、(a) 3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義するとともに、

マッピング対象となる2次元物理プロセッサにおいて、X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化することを前提とし、(b) 3次元計算モデルを $VX=0$ 、 $VY=0$ 、 $VZ=VCZ$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号(メインユニット番号)を付与し、このメインユニットのX方向計算単位数、Y方向計算単位数をvx、vyとし、

2次元物理プロセッサにおいて、 $PX=PCX/vx$ 、 $PY=PCY/vy$ に分割してプロセッサをグループ化し、各分割単位に通し番号(グループ番号)を付与し、

(c) vx=PX、vy=PYの位置に対応して、メインユニットを2次元物理プロセッサのX方向に通し番号順に配置し、そのライン上でのX方向の端部物理プロセッサに来たら、Y方向へ移行し、今度はメインユニット

を逆のX方向に通し番号順に配置し、再度X方向の端部物理プロセッサに来たら、Y方向へ移行し、以後、以上の蛇行によるマッピングを繰り返すことを特徴とする並列計算機。

【請求項30】 請求項25の処理を並列処理することを特徴とする並列計算機。

【請求項31】 請求項21において、さらにデータベース部を有し、前記マッピングで得られた最適マッピング・パターンをこのデータベース部に蓄積し、以後、このデータベースを参照して、計算モデルに適合する最適マッピングを選択し、並列処理をすることを特徴とする並列計算機。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、並列計算機における関し、特に、並列計算機におけるマッピングを効率よく行い、通信の効率化を図る技術に関する。

【0002】

【従来の技術】コンピュータシステムの高速化や大容量化が要求されるに伴って、複数のプロセッサに処理を分散させる分散処理技術が必要となってきた。

【0003】そこで、複数のプロセッサが処理を並列的に行なう並列計算機が提供されている。この並列計算機では、各プロセッサが通信手段を介して互いに通信を行うことで各プロセッサが並列に動作し、複数のプロセッサ全体であるまとまった処理を実行する。これによれば、1つの仕事に対する処理の高速化を図ることができる。

【0004】近年は、このような並列計算機の開発が進み、並列化を行う環境が次第に整い、実用的なアプリケーションが適応可能となってきた。しかし、並列化の過程で必要となる作業、例えばマッピングなどは未だ自動化されていない。特に、マッピングの際のデータ分割や、機能分割のアルゴリズム(以降、分割アルゴリズム)を考える時に、並列計算機のネットワーク・アーキテクチャの構造を考慮に入れなければならない。

【0005】この場合、並列計算機のネットワーク・アーキテクチャの構造が様々であることから、効率のよいプログラムを開発するためには、一般ユーザが、マシンのアーキテクチャを十分に考慮する必要がある。

【0006】このようにして完成した並列アプリケーション・プログラムでは、並列計算機のアーキテクチャが進歩した場合、再度、改良されたアーキテクチャに合わせてプログラムを変更しなければならない。

【0007】従って、事実上は、並列アプリケーション・プログラムは並列計算機のアーキテクチャの変化に追いついて行けないであろう。これでは、適応性の高い並列アプリケーション・プログラムの蓄積は望めない。

【0008】

【発明が解決しようとする課題】このような現状の環境

にもかかわらず、一部の先進ユーザは、アプリケーションの高速化を達成するために、アーキテクチャを十分に考慮して高速化を行っている。

【0009】一方、並列計算機を普及させるには、一般ユーザに分割アルゴリズムを意識させないで、並列アプリケーションを開発させる環境が必要である。それには、使い勝手だけを考えれば、ユーザが自分のN次元計算モデルのイメージのままで並列アプリケーション・プログラムの開発が出来ればよい。また、アプリケーションの高速化という観点では、先進ユーザの計算モデルに対応したマッピングが必要になる。

【0010】そこで、アプリケーションによく見られるデータ参照パターンについて、ユーザに使いやすくかつ、高速であるインターフェースが必要となる。本発明は、このような要請に基づきなされたもので、並列計算機において、ユーザが、並列計算機のアーキテクチャを意識せずに、アプリケーションの開発あるいは運用を図れるようにすることを課題とする。

【0011】**【課題を解決するための手段】**本発明は、前記課題を解決するため、複数のプロセッサの間で相互にデータ及び情報を通信手段により転送することで各々のプロセッサが並列に処理を実行する並列計算機において、以下の構成をとった。

【0012】図1の原理図に示したように、並列計算機において、計算モデルを分割した計算ユニットと、並列計算機のアーキテクチャである複数の物理プロセッサとが、アドレス変換テーブルを介して、対応づけられる。

【0013】図1で、21は、ユーザが演算処理しようとするN次元計算モデルである。ここでは3次元モデルであるが、任意のN次元(N=自然数)が可能である。N次元計算モデルは、識別符号を有する複数の計算ユニットに分割される。

【0014】また、23は、物理プロセッサ構成である。ここでは、2次元に配列されている。複数の物理プロセッサ(PE)には、識別符号が付与されている。25は、分割された複数の計算ユニットと、各プロセッサとを、識別符号により対応づけるアドレス変換テーブルである。

【0015】ユーザが計算させようとする計算モデルは様々であるが、適応する問題にはアプリケーション独特のデータ参照パターンが存在していて、さらに計算モデル構成が3次元以上の高次元であるものが多く、ネットワークアーキテクチャの構成と異なることが多い。前記問題を解決するためには、これを反映させる必要がある。

【0016】これらを踏まえ、ユーザに、アーキテクチャを意識せず、計算モデルのイメージのままで作業させるためには、あたかも、物理プロセッサ構成がユーザの計算モデルと同じ構成をしているかのごとく、ユーザ

にイメージさせて、その空間(ユーザ空間)のイメージのままでプロセッサ間通信をさせるインターフェースを定義しなければならない。このようなインターフェースを実現するのが以上の構成である。

【0017】そして、このユーザ・インターフェースを介して、分割した計算ユニットを、各プロセッサに任意にマッピングする。マッピングとは、計算モデルを、アプリケーションプログラム上で並列に計算できる計算領域の単位(ユニット)に分割したあとのプロセスであり、各ユニットを並列計算機上の、各プロセッサに割り当てることをいう。ここで、前記単位(ユニット)は、アプリケーションプログラムで並列に計算できる計算領域の最小単位ユニットでもよいし、この最小単位ユニットをいくつか集めたユニットブロックであってもよい。

【0018】本発明は、方法として、あるいは、ライブラリ・プログラムとして、さらには、並列計算機自体としてとらえることが可能である。まず、方法としてとらえた場合、並列計算機において、ユーザが演算処理しようとするN次元計算モデルを、前記複数のプロセッサで並列処理する方法であり、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割するステップと、前記分割した計算ユニットと各プロセッサとを対応づけるステップとでユーザ・インタフェースを形成する。このインターフェースを介して、分割した計算ユニットを、各プロセッサに任意にマッピングするステップを有する。その後、プロセッサ間通信を分割ユニットの識別符号を用いて行う。

【0019】この方法をライブラリ・プログラムとした場合、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割する分割ルーチンと、分割した計算ユニットの識別符号と前記各プロセッサに付与された識別符号との対応関係をアドレス変換テーブル上に形成する管理ルーチンと、分割した計算ユニットを、各プロセッサに任意にマッピングするマッピングルーチンと、を備え、プロセッサ間通信を分割ユニットの識別符号を用いて行うことを特徴とする並列計算機用通信ライブラリとして実現できる。これを、従来のライブラリ群に加えて、N次元サブルーチン・ライブラリとする。

【0020】このようなライブラリの実行により、並列計算機において、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割する計算モデル分割部と、前記複数の各プロセッサに識別符号をつけておき、分割した計算ユニットと各プロセッサとの対応関係をアドレス変換テーブル上に形成する管理部と、分割した計算ユニットを、各プロセッサに任意にマッピングするマッピング部と、が実現される。

【0021】以上において、計算モデルに対応した最適なマッピングを行う必要がある。最適なマッピングとは、アプリケーションのデータ参照パターンに対し、高速化という観点で最も効率的なプロセッサ間通信を実現



するものである。

【0022】本来、ユーザ空間から、プロセッサ空間へのマッピング作業は、自動的に最適な形が選ばれるのが理想的である。しかし、アプリケーション独自のデータ参照パターンは、ユーザしか知らないもので、これまで多くの並列計算機システムでは、満足のいくマッピングを行ってはいなかった。また、最適なマッピング手法自体分からない部分が多い。そこで、本発明では、まず各通信パターンに対応した、最適なマッピング規則を見つけることに目標をおいた。そのためには、ユーザが自由にマッピング規則を操作できるものが必要になる。

【0023】前記で説明した構成は、このような要請に応えるユーザ・インターフェースを提供する。マッピングに当たっては、通信の高速化の観点から、ネットワークアーキテクチャに加え、新たにアプリケーションのデータ参照関係を考慮する必要がある。そこで、マッピングの決定という問題を前述のように通信の高速化問題としてとらえ、通信時間の評価関数を設定し、通信の最小化を図ることとした。

【0024】通信時間の最小化のアルゴリズムとして、焼き鈍し法（アニーリング）を採用することで並列計算機上の通信時間が最小となるようにするとよい。代表的なデータ参照関係の場合は、この最適化問題をユーザが解くことなく、最小な時間で、最適なマッピング情報を得られるようにするとよい。

【0025】代表的なデータ参照関係とは、最近接参照、近接参照、全対全、等これまで経験的にアプリケーションプログラムに見られたものである。ユーザ固有のデータ参照関係の場合は、予め最適化問題を解き、最適マッピング情報データベースにこれを登録することで、代表的なデータ参照関係の場合と同様に、アプリケーションプログラムの中からデータベースにアクセスするだけで最適なマッピングを得られるようにするとよい。

【0026】ユーザ固有のデータ参照関係とは、ユーザにきわめて強く依存したもので、前記したデータ参照パターン以外の汎用性にとぼしいものである。例えば、“2つ飛び”、とか、“ある領域のみ”とかいった極めてまれなデータ参照パターンである。

【0027】最適マッピング情報データベースシステムの作成にあたっては、データの参照関係、計算ユニット単位のプロセッサ間データ量、プロセッサ台数を定義し、マッピングの最適化手法により、マッピング情報をデータベース出力する。

【0028】データベースが構築されると、その後は、並列処理に当たって、このデータベースを参照し、計算モデルにふさわしいマッピングを選択して、処理を遂行する。

【0029】

【作用】本発明では、前記図1のような、ユーザ・インターフェースを提供し、ユーザが分割したN次元モデル

の計算ユニットと、物理プロセッサに識別符号をつけておき、その対応をアドレス変換テーブルでもつので、これにユーザがアクセスするだけで、自由にマッピング操作をすることが可能になる。

【0030】その後は、プロセッサ間通信はユーザが分割ユニットの識別符号を用いて行うことが出来る。これで、ユーザ空間での作業が実現し、N次元計算モデルに応じてマッピングを自由に操作できるようになる。

【0031】

【実施例】以下、本発明の好適具体例を図を参照して説明する。実施例では、並列計算機AP1000を対象に行った研究について、このユーザ・インターフェース実現の方法を述べるとともに、典型的なデータ参照パターンについての、最適なマッピング方法について述べる。＜AP1000について＞まず、並列計算機AP1000を、図2、図3に従って説明する。

【0032】AP1000は、各プロセッサが、2次元トラス状に接続されているMIMD (Multi Instruction stream Multi Data stream) 型の並列コンピュータである。特徴として、3つの通信ネットワークを装備していて、プロセッサ (PE) 間の通信はトラスネットワークを用いる。PE間通信の最適化にあたっては、ネットワークの特性を考慮しなければならない。

【0033】図2にAP1000のアーキテクチャ構成図を示す。AP1000は、1対1の通信に使用するトラス・ネットワーク31 (Torus network) (以下、T-net という)、1対多の通信に使用する放送ネットワーク33 (Broadcast network) (以下、B-net という)、バリア同期専用の同期ネットワーク35 (Synchronization network) (以下、S-net という) の3種の独立した通信ネットワークをもつ。

【0034】T-net はその2次元格子点にルーチング・コントローラ37 (RTC) を有し、B-net は複数の放送ネットワーク・インターフェース39 (BIF) を有している。T-net のRTC37とB-net のBIF39とは4対1の関係でそれぞれバスで接続され、各バス上に、それぞれセル・プロセッサ41 (以下、単にセルという) が設けられている。その内にいくつかには、オプション構成として、フレーム・バッファ43と、ハード・ディスク45とが接続されている。また、フレーム・バッファ43にモニタ47が接続されている。

【0035】前記B-net の各BIF39は、S-net35に一括して接続されている。また、BIFの一つにホストコンピュータ49が接続されている。この結果、全セル41とホスト49は、B-netによって接続される。

【0036】B-netは、階層バスとリングを組み合わせたネットワークで、データ放送、分散、収集に使用する。B-netは、データ転送中は一つのセルまたはホストに占有されるので、データ転送を行いたいホストまたはセルは、データ転送を行う前にB-netの使用要求を出し



使用権を獲得する。B-netは、32-bitのデータバスとリセット、割込みなどの制御信号から構成されており、パイプライン化されたハンドシェーク制御によって、50MByte/sのデータ転送レートをもつ。

【0037】T-netは、2次元トラス状のトポロジーをもつネットワークでメッセージの中継は、ワームホールルーチングによりハードウェアで自動的に行われる。ワームホールルーチングでは、メッセージのヘッダが、入力チャネルから出力チャネルへ中継ルートを作りながらメッセージを送り出す。ストアアンドフォワードルーチングが、中継プロセッサがメッセージ全体をストアするのにに対し、ワームホールルーチングではフリットと呼ばれる数バイト（ビット、AP1000では16ビット）のデータのみが中継プロセッサにストアされるため、低レイテンシが実現できる。

【0038】このように、ワームホールルーチングは、通信の遅延時間（レイテンシ：Itency）が小さいという特長をもつが、メッセージが通信チャネルをブロックするためデッドロック発生の可能性とスループットの低下という問題がある。AP1000のT-netでは、ワームホールルーチングに構造化バッファプールアルゴリズムを組み合わせることでデッドロックを回避しスループットの低下を抑えた。また、放送通信のレイテンシを小さくするため、行または列への放送機能も同時にインプリメントした。T-netのそれぞれのポートは、16bit幅のデータバスをもち、B-netと同じようにパイプライン化されたハンドシェーク制御により、25MByte/sのデータ転送レートをもつ。

【0039】全セルとホストは、S-netによっても接続されている。S-netは、ツリー状のトポロジーをもち、バリア同期とステータスの検出に使用される。各セルからのデータは、S-netの根の部分に向かって送られる。S-netの各ノードでは、各セルからのデータがAND演算によってマージされる。全セルの出力したデータの理論積がS-netの根の部分で得られる。得られた結果は、今度はS-netを逆に進んで全セルに知らされる。各セルがS-netに送り出したデータの結果が得られるまでの時間は、8クロック（640ns）とセル数に関係なく一定である。

【0040】ホストコンピュータには、汎用ワークステーションを使用している。AP1000のホストインターフェースは、ホストコンピュータに実装するVMEバスインターフェースボードとAP1000フレーム内に実装されるホストインターフェースボードから構成される。ホストインターフェースボードには、B-netインターフェースと32MByteのローカルメモリが実装され、メッセージバッファとして利用できる。

【0041】図3に前記セルの構成を示す。個々のセルは、整数演算ユニット51（IU）、不動小数点演算ユニット53（FPU）、メッセージコントローラ55

（MSC）、ルーチングコントローラ57（RTC）、B-netインターフェース59（BIF）、16MByteのメインメモリ（DRAM）61から構成される。IUとFPUは、128KByteのダイレクトマップキャッシュメモリ63に接続され、25MHzで動作する。IUには、SPARCアーキテクチャのものを採用した。RTCには、2次元トラスネットワーク上での自動ルーチング機能を、BIFには、データ分散収集及びバリア同期機能をインプリメントした。MSC、RTC、BIF、とDRAMコントローラ65（DRAMC）は、LBUSとよばれる32bitの内部バスで接続される。個々のセルのLBUSは、コネクタを介して外部に取り出されており、高速I/Oインターフェース、拡張メモリ、ディスクインターフェース、ベクタープロセッサ等の個々のオプション・ハードウェアが接続できるようになっている。

【0042】メインメモリは4重インタリーブでDRAMを制御するDRAMCと、40個の4M-DRAMで構成される。MSCはキャッシュコントローラ、ラインセンド（Line sending）とバッファレシブと呼ばれる1対のメッセージハンドラー、4チャンネルの高機能DMAコントローラから構成される。

【0043】本実施例では、このような構成のアーキテクチャ上に、図4のような、前記N次元計算モデルを識別符号を有する複数の計算ユニットに分割する計算モデル分割部71と、前記複数の各プロセッサに識別符号をつけておき分割した計算ユニットと各プロセッサとの対応関係をアドレス変換テーブル上に形成する管理部73と、分割した計算ユニットを、各プロセッサに任意にマッピングするマッピング部75と、が実現される。そして、マッピング部75で得られた最適マッピング情報はデータベース部77に出力される。このデータベース部77は、図2のハードディスク45上に設けられる。以後は、このデータベース部77を管理部73を介して参照し、アドレス変換テーブルを構成して、並列計算を行う。

【0044】ところで、ユーザ空間の重要性、また、そのユーザ空間からの自由なユーザ・マッピングの必要性は前記したところであるが、このような要請に応えるインターフェースをAP1000上に実現した。

【0045】実現方法の概要は、図4で示したように、最初に、ユーザが分割したN次元モデルの計算ユニットと、物理プロセッサに識別符号をつけておき、その対応をアドレス変換テーブルでもつ方法である。すると、これにユーザがアクセスするだけで、自由にマッピング操作をすることが可能になる。その後は、プロセッサ間通信はユーザが分割ユニットの識別符号を用いて行うことができる。これで、ユーザ空間での作業が実現し、N次元計算モデルに応じてマッピングを自由に操作できるようになる。これを、従来のライブラリ群に加えて、図5

で示したようなN次元サブルーチン・ライブラリとして拡張、実現した。

【0046】より詳細には、使用するプロセッサ数を宣言するステップ、宣言したプロセッサの数の範囲内の数 $=N^d$ に計算モデルを計算ユニットへと分割するステップ、分割した計算ユニットに識別符号を付与するステップ、使用するプロセッサに識別符号を付与するステップ、計算ユニットに付与された識別符号と、プロセッサに付与された識別符号とを対応づけて管理テーブルに登録するステップとによりユーザ・インターフェースが構築される。なお、分割された計算ユニットと、プロセッサに付与される識別符号は、1、2、3・・・というような通し番号が管理上好ましい。

【0047】＜処理の流れ＞最適マッピングをするためには、図6のように、まず、計算モデルについての情報を入力する必要がある。すなわち、データの参照関係（データ参照パターンともいう）、計算ユニット単位のプロセッサ間データ量、プロセッサ台数を定義する（ステップA1）。

【0048】そして、マッピングの最適化手法により、最適マッピングを得る（ステップA2）。マッピングの最適化手法については後記する。得られたマッピング情報をデータベース出力する（ステップA3）。

【0049】＜アプリケーションプログラムでの使用例＞データベースが構築されると、その後は、アプリケーションプログラムなどにおいて計算モデルの並列処理に当たって、以下のような処理が行われる。まず、図7で示したように、ユニット分割次元数とマッピングテーブルの入力が行われる（ステップB1）。ここで、マッピングテーブルの入力とは、前記したユーザ・インターフェースの構築である。次に、最適マッピング情報データベースにアクセスして、計算すべき計算モデルにふさわしい最適マッピング情報を読み出し、システム環境を設定する（ステップB2）。ここでは、図4のアドレス変換テーブル25上にマッピング・テーブルが構築される。

【0050】この処理が終了すると、ユニット分割次元上での通信を行いつつ、並列的に処理が行われる。＜計算モデルに対応したマッピング＞では、最適なマッピングを得る方法について説明する。

【0051】マッピングは、図8に示したように、データを任意のN次元に分割したあと、これを最適にプロセッサに割り当てることである。図9は、2次元に分割すべきか、3次元に分割すべきか、マッピングとしてパターン1がよいのかパターン2がよいのかという検討すべきことを示している。

【0052】マッピングに当たって考慮すべきことは、データ参照パターン、計算モデルの次元数、通信コストの評価関数である。

「データ参照パターン」データ参照パターンと通信と計

算のバランスを考慮して、分割の仕方が決定される。

【0053】マッピングの前提としてデータ分割が問題となるが、考慮すべきことは、分割によって分割ユニット間での通信がどうなるかということと、ユニット内の計算量を他ユニットとの通信量とのバランスである。

【0054】例えば、あるLattice sizeの立体格子モデルがあり、これを分割して各セル（プロセッサ）に割り当てるとき、データ参照パターンが最近接（±X方向、±Y方向、±Z方向の合わせて6方向を参照する）であるとしたら、AP1000は2次元トラス構造なので、マッピングの容易性を考えて図10（A）のように分割する。これを2次元分割という。ところが、このような分割では問題がある。すなわち、2次元分割では、図11（A）のように、通信方向が4方向となり、立体モデルで表面積を通信量、体積を計算量と見るとき、分割数が増えていくと、すなわち、セル（プロセッサ）台数を増やしていくと、計算量に対して通信量の占める割合が増えてゆく。

【0055】そこで、図10（B）のような分割方法を考慮してみた。これを3次元分割という。3次元分割では、図11（B）のように、通信方向が6方向となり、最近接のデータ参照パターンに対応できる。このため、通信量の占める割合が2次元分割に比較して少なくなる。しかし、マッピングの仕方が難しくなる。

【0056】以上をまとめると、単位通信方向の通信量の占める割合を計算量に対して減らすために、分割の方法を考え、分割によって発生した通信パターンを考慮にいれマッピングの容易さを考慮するということである。

【0057】ところで、データ参照パターンは、アプリケーションによってさまざまである。最初から全ての場合について考えるのは無理であろう。そこで、図9のように、アプリケーションによく見られるデータ参照パターンによって分類してみることにする。

最近接格子：自分のデータ更新に必要なデータを最も近い隣の格子点から参照する必要がある格子である。±X方向、±Y方向、±Z方向の合わせて6方向を参照する。

【0058】応用例として、構造解析、熱伝動、流体解析（有限要素法、中心差分）、物性（イジングスピ）、MD（モデキュラダイナミクス）などの計算に利用できる。

近接格子：自分のデータの更新に必要なデータを近くで隣合っている（最近接+斜め方向の）格子点から参照する必要がある格子をいう。±X方向、±Y方向、±Z方向の合わせて6方向に加えて、±ZX方向、±YZ方向、±XY方向の合わせて6方向、合計12方向を参照する。

【0059】応用例として、QCD（Quantum chromodynamics：量子色力学）などに利用できる。

N対N：データの更新に全ての格子点を参照する必要

があるもので、応用例として、希薄流体の粒子追跡などに利用できる。希薄流体の粒子を追跡するとき、どの領域に粒子が行くか分からないので、全ての分割領域を参照する必要がある。

完全独立：通信が無い場合である。この場合、計算が並列計算機の全てのプロセッサ（PE）で独立に行える場合である。

【0060】以上の各場合と、分割すべき次数との関係を図9に示す。以上のデータ参照パターンの相違により本発明でどのような影響があるかを、前記3次元分割の場合を例にとりて説明する。

【0061】最近接のとき：3次元分割によって各ユニットには6方向の通信が発生する。これを2次元のセル（プロセッサ）へマッピングするのであるが、2次元セルには4方向の通信経路しか存在しない。AP1000では、セル間の距離に通信時間は依存しないのが建て前であるが、メッセージの競合を避けるためには互いに通信すべきユニットはできるだけ近くに置きたい。そこで、4方向を隣に置けるが、残りの2方向をどのように処理するかが問題となる。

【0062】近接（斜め方向も含む）のとき：この場合、ユニットが計算に必要とする方向が増えるのでその分、マッピングの仕方に影響がでる。

N対Nの場合：この場合は、データの更新に全ての計算単位の参照が必要な場合であるから、その点を考慮する必要がある。この場合、分割によって膨大な通信量が発生するのでマッピングには近接と異なる考え方を導入する必要がある。

「計算モデルの次元数とマッピング」データの参照は、問題が何次元であるかによっても考え方が異なってくる。例えば図10、図11の様に、データ参照が最近接方向で2次元の計算モデルを、AP1000のような2次元トラスにマッピングする場合は、そのままマッピングすればいいが、3次元の計算モデルの場合だと、途中で変換が必要になる。計算モデルの次元が増える程、各計算ユニットが通信する方向は増えてゆき、高次元の計算モデルになるにつれて、難しくなることが想像される。例えば、3次元の最近接格子の場合は通信方向が6方向で、4次元だと8方向である。これを2次元にマッピングするには、残りの方向に当たる部分をどこかに持ってゆかなければならない。

【0063】実施例では、AP1000（2次元トラス）での3次元の最近接格子の通信の最適化問題を考えることとした。これは、アプリケーションで最も応用範囲の広いケースである。

「通信コストの評価関数」一般的に、並列計算機で通信コスト（時間）の削減を考える時に、通信回数、通信量、通信の混み方、プロセッサ（PE）間距離を考えねばならない。どれが、どのくらい通信コストに、効いてくるかの比率は、その並列計算機のアーキテクチャに大

きく依存している。AP1000の場合は、プロセッサ間通信コストは、図12、図13に示されるように、プロセッサ（PE）間の距離にはあまり依存せず、通信の混み方に大きく依存している。

【0064】しかし、通信が混んでくると、むしろ通信距離ができるだけ少ない方が、通信の衝突も起こりにくく混み方が少なくなり、通信コストが最小に近い値が得られるだろうと仮定した。

【0065】このことは、図14のAP1000のネットワーク特性からも想定できる。図14の実験は、全てのセルが同じ方向、同じ距離だけ離れたセルに一斉に送信した場合メッセージの発信から受信までどれくらいかかるか？という実験である（メッセージの交換ではない）。これは $32 \times 16$ セルなのでトラスの効用最長送信距離（X，Y）は図から明かなように24となる。この結果から、データが大きくなり、経路上でメッセージの引きずりが起こるとき距離に対してリニアに通信時間が上がるように見える。そこで、これを確認するために、評価関数として各プロセッサ間の距離の総和Lを導入した。マッピングは、この距離の総和Lが最小となるように行う。AP1000について、距離の総和と通信速度との関係を調べたところ、図15のように、距離の総和が小さいほど、通信速度が速いことがわかった。

「マッピング」実施例では、サンプルとして、 $3^3$ の物理体系を $8^2$ の2次元ネットワーク構成の計算機に、 $4^3$ を $8^2$ にマッピングする場合について考える。しかし、全ての組み合わせはそれぞれ、 $10^{48}$ 通り、 $10^{90}$ 通り存在するため、これら全てを調べるのは無理である。そこで、経験的に規則的な並びを考える方法と、シミュレーションにより求める方法を行った。

#### \*経験的マッピング

3次元の計算ユニットの、6つの通信方法全てに対し均等な通信コストは考えにくいので、まず、ある方向を優先的に隣になるようにおき、残りの方向についてなるべく近く、規則的になるように置くことを考えた。

【0066】XYZ各方向に対して、立体的に分割し、それをマッピングするモデルを考える。図16は $3^3$ を $8^2$ にマッピングした場合であり、図17は $4^3$ を $8^2$ にマッピングした場合である。図16、図17に於いて、Aのマッピング方法（スキップ法という）は、ZX平面で4つに切っておき、Y方向の各計算ユニットが正方形で隣合うように並べる方法で、ZX方向の通信がひとつ飛びに並ぶ恰好になる。しかし、Y方向が隣合う為には、ZX平面を4つに分割しなければならない。

【0067】より具体的には、（a）3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義する。同時に、マッピング対象となる2次元物理プロセッサにおいて、X方

向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化する。

【0068】そして、(b) 3次元計算モデルを $VX=PCX/2$ 、 $VY=PCY/2$ 、 $VZ=0$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号(メインユニット番号)を付与し、さらにこのメインユニットをZ軸方向に4つに分割し、この結果できたユニットをサブユニットと呼んで、各分割単位に通し番号(サブユニット番号)を付与し、同時に、2次元物理プロセッサにおいて、 $PX=PCX/2$ 、 $PY=PCY/2$ に分割してプロセッサをグループ化し、各分割単位に通し番号(グループ番号)を付与する。

【0069】さらに、(c) 3次元モデルのメインユニット番号と同一番号のグループ番号を有する物理プロセッサのグループに、図18のように、サブユニット番号0から3を、0と1、1と2、2と3、3と0とが隣接するように分配してマッピングする。図16、図17に於いて、Bのマッピング方法(タイル法という)は、ZX平面でスライスしたものを順番に置いていったものである。これで、ZX方向が隣合いY方向が均等(スライス数分離れる)に並ぶ。

【0070】より具体的には、(a) 3次元計算モデルにおいて、X方向、Y方向、Z方向の計算単位をVCX、VCY、VCZと定義し、X方向、Y方向、Z方向の分割数をVX、VY、VZと定義する。同時に、マッピング対象となる2次元物理プロセッサにおいて、X方向物理プロセッサ台数、Y方向物理プロセッサ台数をそれぞれPCX、PCYと定義するとともに、X方向、Y方向に分割した分割数をそれぞれPX、PYと定義して物理プロセッサをグループ化することを前提とする。

【0071】そして、(b) 3次元計算モデルを $VX=0$ 、 $VY=0$ 、 $VZ=VCZ$ に分割し、この分割によって出来た直方体ユニットをメインユニットと呼び、各分割単位に通し番号(メインユニット番号)を付与し、このメインユニットのX方向計算単位数、Y方向計算単位数をvx、vyとする。同時に、2次元物理プロセッサにおいて、 $PX=PCX/vx$ 、 $PY=PCY/vy$ に分割してプロセッサをグループ化し、各分割単位に通し番号(グループ番号)を付与する。

【0072】さらに、(c)  $vx=PX$ 、 $vy=PY$ の位置に対応して、図19のように、メインユニットを2次元物理プロセッサのX方向に通し番号順に配置し、そのライン上でのX方向の端部物理プロセッサに来たら、Y方向へ移行し、今度はメインユニットを逆のX方向に通し番号順に配置し、再度X方向の端部物理プロセッサに来たら、Y方向へ移行し、以後、以上の蛇行によるマッピングを繰り返す。

\*シミュレーションによるマッピング

次に、シミュレーションによって最適なマッピングを求めようという方法を用いた。つまり、今回のマッピングの問題を距離の総和を最小化する組合わせの最適化問題として考えた。

【0073】マッピングによる通信コストは、すべての最近接間格子間の距離の総和にほかならないとした。方法は、初期状態としてランダムにマッピングしておき、経路の総和を求めておく。図20のように、どこか一組を取り替えてみて、経路の総和が減ったら交換を採用する。交換を繰り返していき、総和の変化が少なくなった状態を近似解として採用する方法である。

【0074】具体的には、隣接格子点 $P_i P_j$ がそれぞれ2次元格子点上のある点 $(X_i, Y_i)$   $(X_j, Y_j)$ にマッピングされたとする。 $P_i P_j$ の距離 $l_{ij}$ は

$$l_{ij} = |X_i - X_j| + |Y_i - Y_j|$$

よって評価関数は、 $L = \sum l_{ij}$ になりこれが最小になった時の、マッピング状態を最適なものと近似する。

(最小なのは全ての格子点が一つになったときであるが、交換で格子点が重なることは考えない)ところが、このような緩和法には、局所的最小の存在が確認されており、 $\Delta L$ が最小だからといって必ずしも、通信コストが最小であるとは限らない。

\*アニーリング法

そこで、組合わせの最適化の近似解法として、カークパトリックら(S. Kirkpatrick et al. 1983)によって提案された、シミュレーテッドアニーリング法(simulated annealing method) [4]を用いた。これは、評価関数をより小さくするように変化させていく過程に、局所的最小を脱出するような確率を導入した方法である。その確率を $\omega$ 、目安をT(確率を決定するパラメータ)とする。

【0075】

$$\omega(\Delta L) = \exp(-\Delta L/T), \Delta L > 0$$

$$\omega(\Delta L) = 1, \Delta L < 0$$

$\Delta L = L(\text{交換前}) - L(\text{交換後})$ : 距離の変化

T(確率を決定するパラメータ)を段々下げてゆく(確率を下げてゆく)と評価関数が最小に近くなる。Tの下げかた(アニーリング・スケジュール)は、ギーマン兄弟(S. Geman and D. Geman 1984)による、

$$T(t) = B / \ln t$$

$$T(t) \rightarrow 0 \quad (t \rightarrow \text{無限大}) \text{とした。}$$

【0076】B: 評価関数の障壁の高さ

図21に、アニーリング法のアルゴリズムを示す。すなわち、初期マップ位置を決定した後(ステップ101)、マップ位置を変化させ(ステップ102)、長さの変化(前記距離の総和の変化) $\Delta L$ を算出し(ステップ103)、 $\Delta L < 0$ のとき、前記マップ位置の

変更を採用し（ステップ104、105）、 $\Delta L < 0$ でないとき、乱数 $X$ （ $0 < X < 1$ ）を引き出し（ステップ106）、 $\exp(-\Delta L/T)$ を演算し（ $T$ は確立を決める目安となるパラメータであり、例えば自然数を代入できる）（ステップ107）、 $\exp(-\Delta L/T) < X$ のとき、マップ位置の変更を採用せず（ステップ108）、 $\exp(-\Delta L/T) < X$ でないとき、マップ位置の変更を採用することとし（ステップ105）、以上からの処理を、前記 $T$ の値を減少させて（ステップ109）繰り返すことにより、最終的に最適なマッピングを得ることができる。からまでの繰返し回数は、経験則から、100万回程度で行うと、 $L$ が十分に下がることがわかった。よって、ループの繰返し回数を予め入力して最適な $L$ を求めることができる。なお、前記が一定回数連続して繰返されたとき、前記処理ループを中断させるブレイクポイントを設けてもよい。

【0077】このアニーリング法でどのように通信時間が減るかを、図22に示す。

＜結果＞このアニーリングによって得られた、マッピング結果、 $3^3 \rightarrow 8^2$ の場合を図23、 $4^3 \rightarrow 8^2$ の場合を図26に示す。

（図中の番号はXYZ各座標を表す）同時に、前記した経験マッピングのAパターン方法（スキップ法）、Bパターン方法（タイル法）についてもマッピング結果を得た。 $3^3 \rightarrow 8^2$ の場合をそれぞれ図24、図25、 $4^3 \rightarrow 8^2$ の場合をそれぞれ図27、図28に示す。

＜マッピングの評価＞図29に示したように、シミュレーションで、得られたマッピングと経験的に得られたマッピングとを距離の総和（ $L$ ）で比較してみた。

【0078】 $3^3 \rightarrow 8^2$ の場合は、経験的に得られたマッピングより、本手法でのマッピングの方が距離の総和が少なく良い結果が得られた。ただし、最良なマッピングであるとは限らない。 $4^3 \rightarrow 8^2$ の場合は、経験的に得られたマッピングと同程度の結果が得られた。

＜データベースへの格納＞以上のマッピングで得られた最適マッピングに関する情報は、図30のようなファイル形式でデータベースに登録される。

【0079】このデータベースがどのように使用されるかは、前記したように、図7で示した通りである。

＜結論＞評価関数として距離をとり、3次元格子の最近接参照モデルを2次元のトーラス・ネットワークにマッピングする時、アニーリング法を用いる事で、結果を比較的容易に経験のマッピングより良い結果が得られる場合があることを示した。今後、評価関数を正確化し改良することで、経験的手法が働かないような、高次元の問題のマッピングを考えるのに有効と考えられる。

【0080】なお、本実施例は、AP1000について行ったが、本発明により前記ユーザ・インターフェースを実現できる並列計算機であれば、他の並列計算機でも

本発明を実施できることはいうまでもない。

＜マッピング最適化の並列処理＞図31に、以上説明したマッピングの最適化を並列処理で行った場合の例を示した。

【0081】ここでは説明を簡単にするため、2台のプロセッサで並列処理を行ったものとする。まず、親プロセッサがマップ位置等の初期値を子プロセッサに放送する（ステップ101-1）。また、親プロセッサでは、障壁サンプルの設定を行い（ステップ206）、障壁 $B$ を子プロセッサに送信する（ステップ207）。このステップ206、207を繰返し、最終フラグを子プロセッサに送信する（ステップ208）。最終フラグとは以下のループを行うか行わないかのフラグである。

【0082】子プロセッサでは、初期値を受信して初期設定を行う（ステップ101-2）。ついで、前記障壁 $B$ を受信し（ステップ201）、マップ位置を変化させる（ステップ102）。ついで、長さの変化（前記距離の総和の変化） $\Delta L$ を算出する（ステップ103）。 $\Delta L < 0$ のとき、前記マップ位置の変更を採用する（ステップ104、105）。 $\Delta L < 0$ でないとき、乱数 $X$ （ $0 < X < 1$ ）を引き出し（ステップ106）、 $\exp(-\Delta L/T)$ を演算する（ $T$ は確立を決める目安となるパラメータで、例えば自然数を代入できる）（ステップ107）。 $\exp(-\Delta L/T) < X$ のとき、マップ位置の変更を採用しない（ステップ108）。 $\exp(-\Delta L/T) < X$ でないとき、マップ位置の変更を採用する（ステップ105）。その後、 $T$ の値を減少させ（ステップ109）、減少率が十分低いかなかを判定し（ステップ203）、目標値に達していない場合、ステップ201から109までの処理を繰り返す。

【0083】ステップ201で最終フラグを受信した場合は、ステップ102から202の処理を飛ばして、全サンプル中での最小値を計算する（ステップ203）。そして、全サンプル中で「PE間距離の総和」が最小であるかなかを判定し（ステップ204）、全プロセッサの中で最小であればその結果を親プロセッサに送信する（ステップ205）。親プロセッサでは、最小マッピングの出力をして（ステップ210）、処理を終了する。なお、子プロセッサのステップ204で全サンプル中で最小でないとされた場合、結果は親プロセッサに送信せず、処理を終了する。ここで、全サンプルとは、全セルでのそれぞれのサンプル、すなわち、あるセルでの評価関数が最小となったマッピングパターンである。

【0084】この並列化処理により、マッピングの最適化が高速で行われる。

【0085】

【発明の効果】本発明では、ユーザ空間において、並列計算機のアーキテクチャを意識することなく最適なマッピングをすることができる。よって、並列処置の高速化を簡単に実現できる。

## 【図面の簡単な説明】

【図1】 本発明で構成されるユーザ・インターフェースを示す概念図  
 【図2】 実施例で使用した並列計算機のアーキテクチャを示す構成図  
 【図3】 実施例のセル構成図を示した構成図  
 【図4】 インターフェースと本発明の機能ブロックとの関係を示した図  
 【図5】 ライブラリプログラムの一部を示した図  
 【図6】 本実施例における処理フロー図  
 【図7】 得られたデータベースの使用例を示す処理フロー図  
 【図8】 データ分割とマッピングパターン選択の考え方を示した図  
 【図9】 データ参照パターンと分割次元数との関係を示す図  
 【図10】 データ分割・マッピングの関係の一例を示した図  
 【図11】 データ分割・マッピングの関係の一例を示した図  
 【図12】 AP1000におけるPE間距離対通信時間を示す図  
 【図13】 AP1000における通信コンテンション対通信時間を示す図  
 【図14】 AP1000における距離と時間との関係を示す図  
 【図15】 AP1000における距離の総和と通信時間との関係を示す図  
 【図16】  $3^3$ を $8^2$ にマッピングした場合を示す図  
 【図17】  $4^3$ を $8^2$ にマッピングした場合を示す図  
 【図18】 マッピング例を示す図  
 【図19】 マッピング例を示す図  
 【図20】 最適マッピングの検索例を示す図  
 【図21】 シミュレーテッド・アニーリング法のアルゴリズムを示す図  
 【図22】 評価関数（距離）による通信時間の差を示す図  
 【図23】  $3^3$ を $8^2$ にアニーリング法でマッピングした場合の具体的結果を示す図  
 【図24】  $3^3$ を $8^2$ にスキップ法でマッピングした場合の具体的結果を示す図

【図25】  $3^3$ を $8^2$ にタイル法でマッピングした場合の具体的結果を示す図

【図26】  $4^3$ を $8^2$ にアニーリング法でマッピングした場合の具体的結果を示す図

【図27】  $4^3$ を $8^2$ にスキップ法でマッピングした場合の具体的結果を示す図

【図28】  $4^3$ を $8^2$ にタイル法でマッピングした場合の具体的結果を示す図

【図29】 マッピングの手法による評価関数と実測の通信時間との関係を示した図

【図30】 最適マッピング情報データベースのファイル形式を示した図

【図31】 マッピングの最適化を並列処理で行った場合のフローチャート図

## 【符号の説明】

21・・・N次元計算モデル  
 23PE・・・物理プロセッサ  
 25・・・アドレス変換テーブル  
 31・・・トーラス・ネットワーク  
 33・・・放送ネットワーク  
 35・・・同期ネットワーク  
 37・・・ルーチング・コントローラ  
 39・・・放送ネットワーク・インターフェース  
 41・・・セル・プロセッサ  
 41・・・全セル  
 43・・・フレーム・バッファ  
 45・・・ハード・ディスク  
 47・・・モニタ  
 49・・・ホストコンピュータ  
 51・・・整数演算ユニット  
 53・・・不動小数点演算ユニット  
 55・・・メッセージコントローラ  
 57・・・ルーチングコントローラ  
 59・・・インターフェース  
 63・・・ダイレクトマップキャッシュメモリ  
 65・・・DRAMコントローラ  
 71・・・計算モデル分割部  
 73・・・管理部  
 75・・・マッピング部  
 L・・・距離の総和

【図1】

【図2】

【図3】

【図5】

ライブラリプログラムの一部を示した図

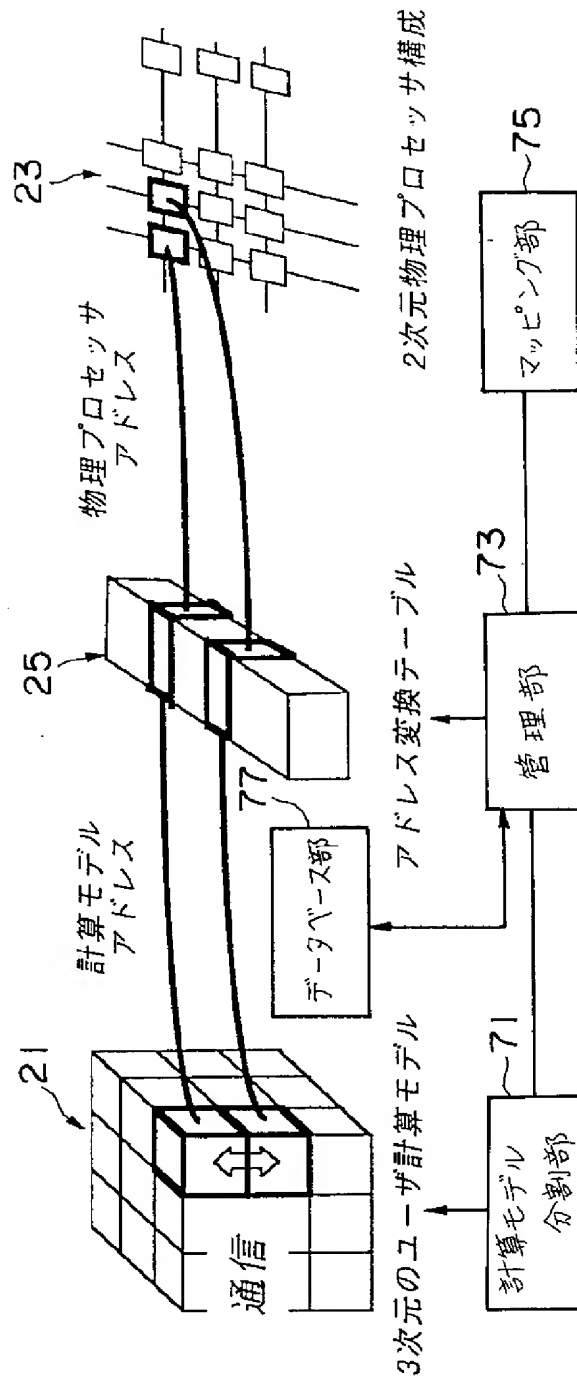
N次元通信ライブラリ使用例

CALL CONF(3次元,	.....)	→	3次元ユーザ空間を定義
CALL SEND(2,0,0,	.....)	→	3次元ユーザ空間上で データの送受信を行う
CALL RECV(2,0,1,	.....)		



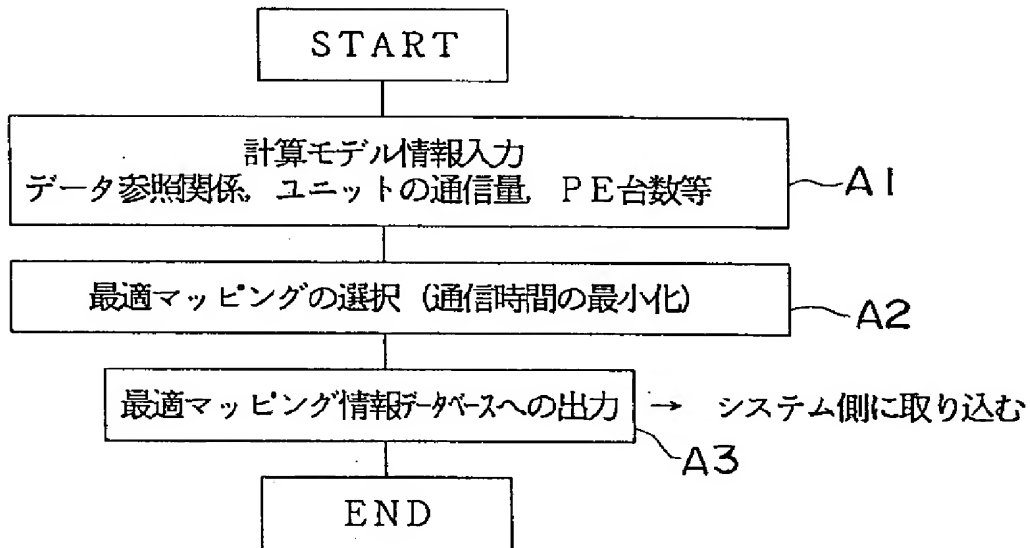
【図4】

インターフェースと本発明の機能ブロックとの関係を示した図



【図6】

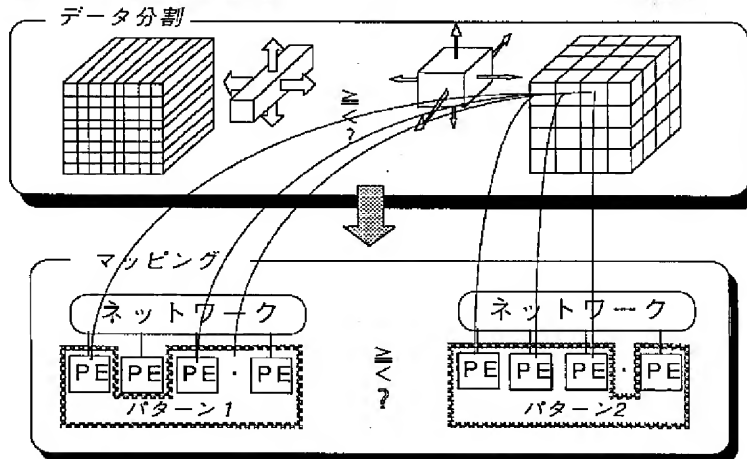
本実施例における処理フロー図



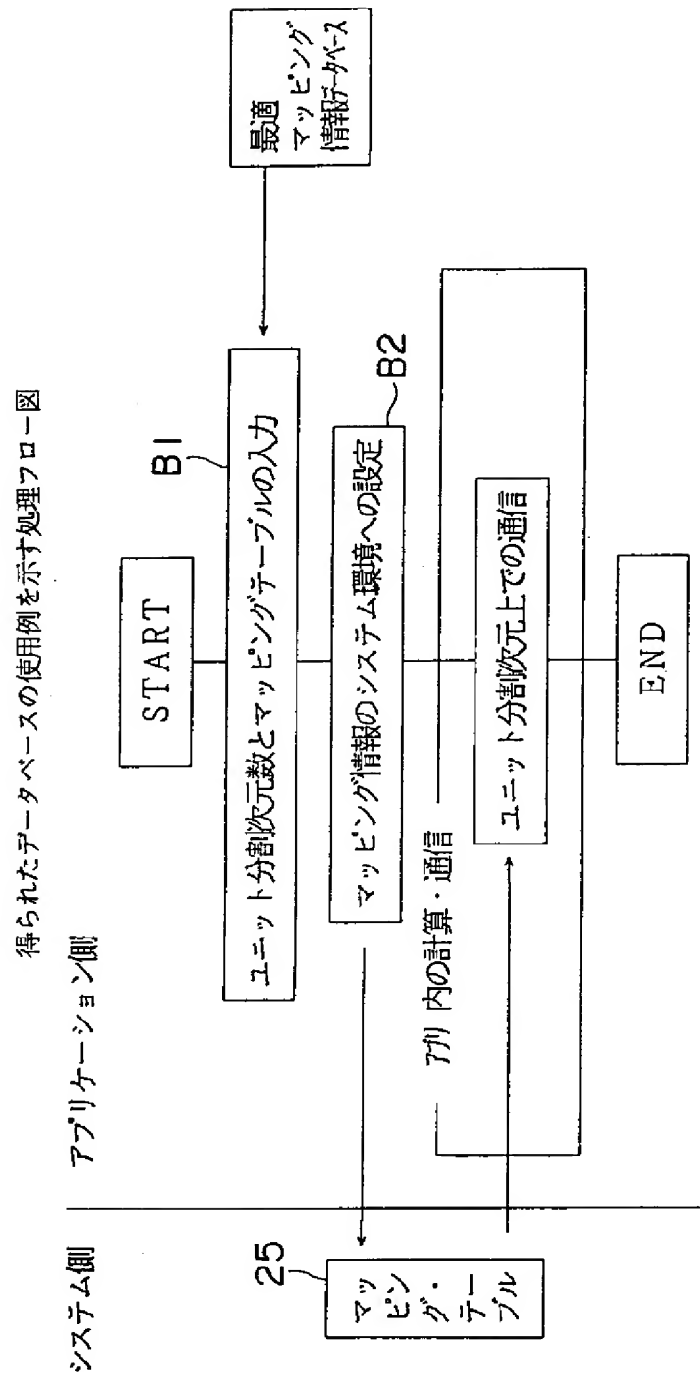
【図8】

データ分割とマッピングパターン選択の考え方

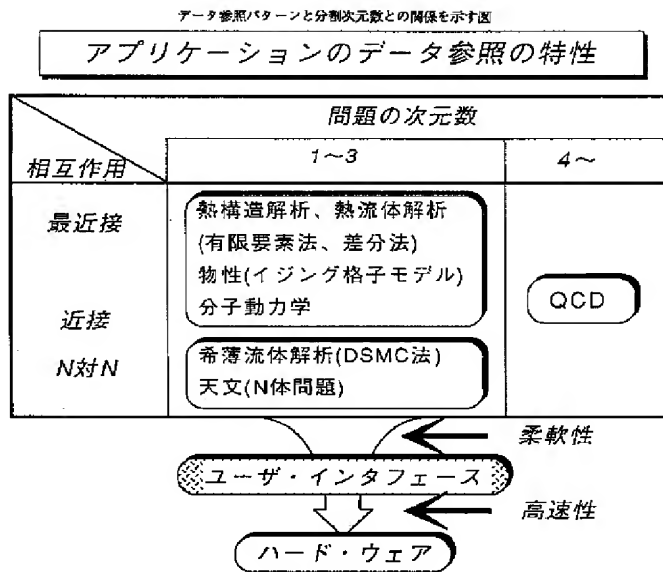
通信時間 = F (通信量、通信回数、アーキテクチャパラメータ (PE間距離、...))



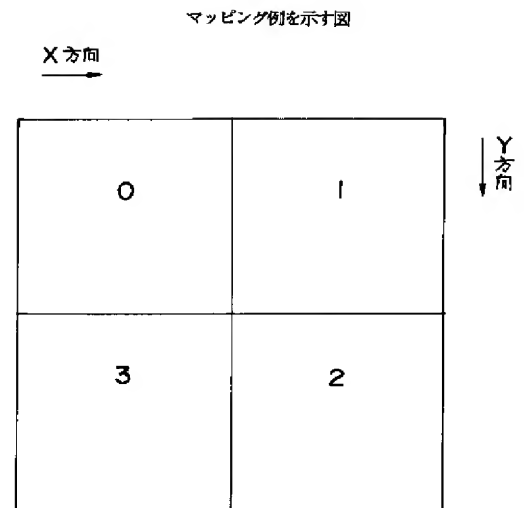
【図7】



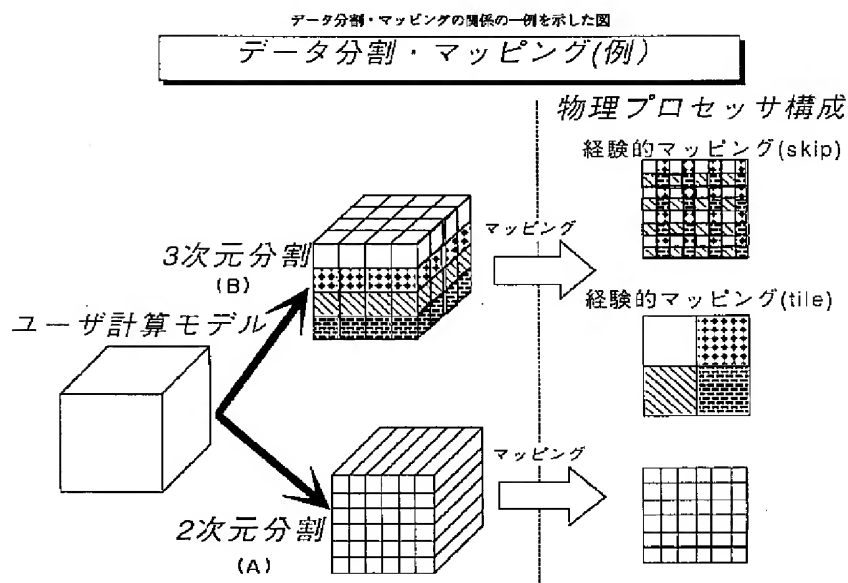
【図9】



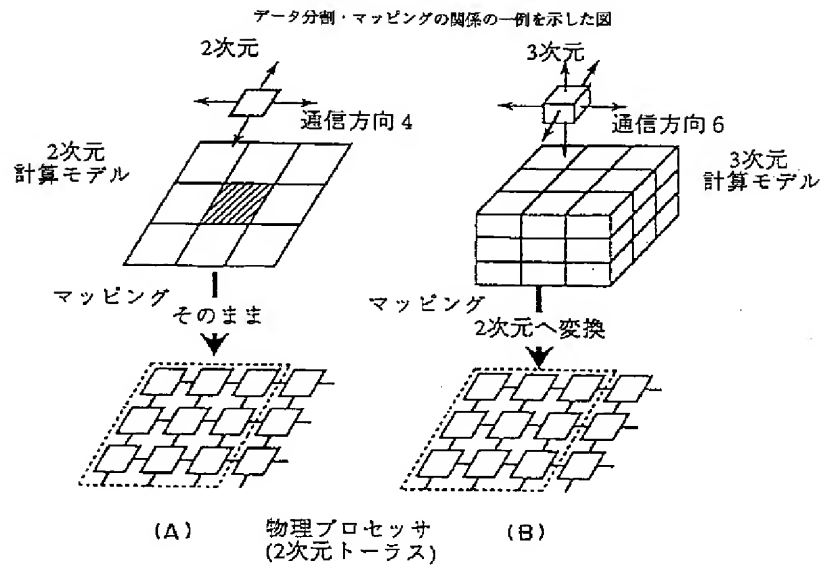
【図18】



【図10】

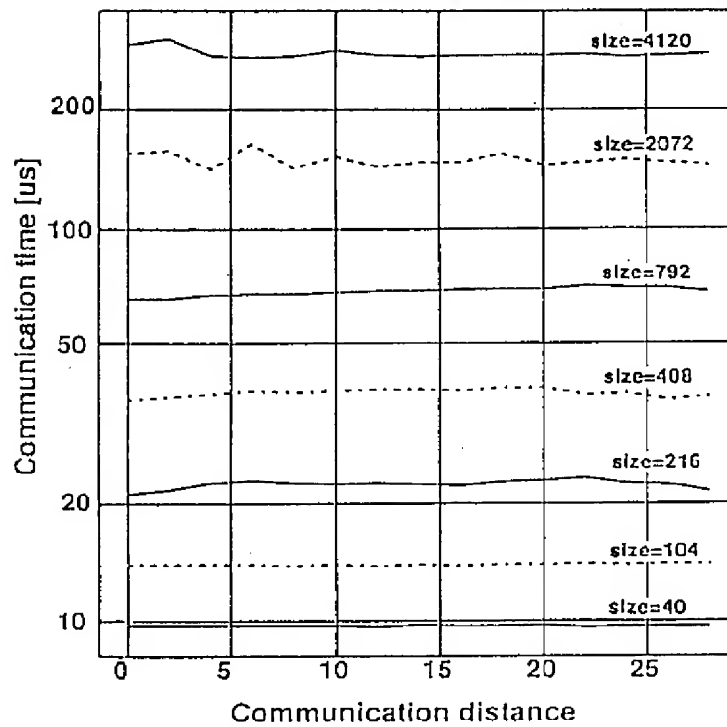


【図11】



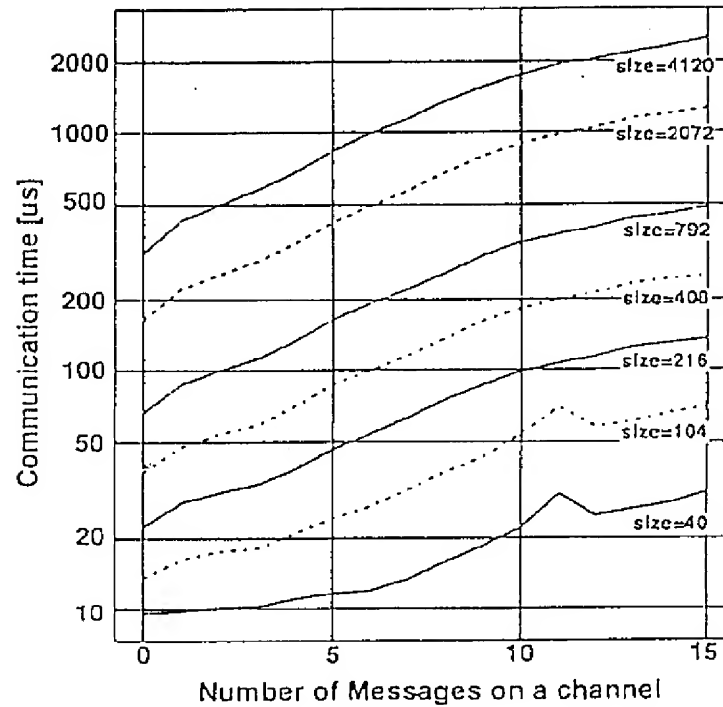
【図12】

AP1000におけるPE間距離対通信時間を示す図



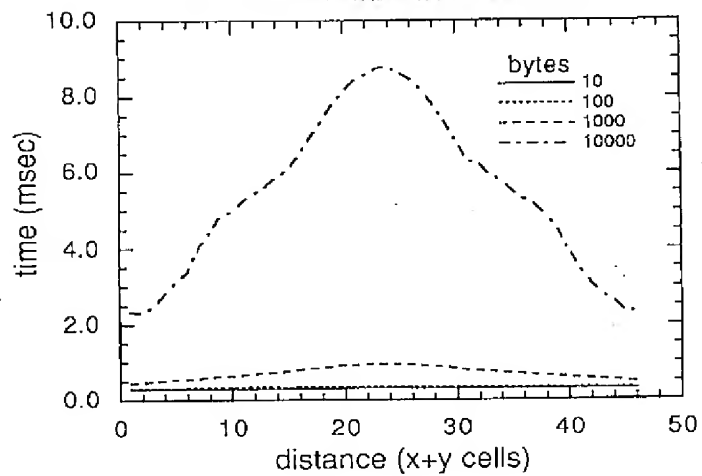
【図13】

AP1000における通信コンテンション対通信時間を示す図

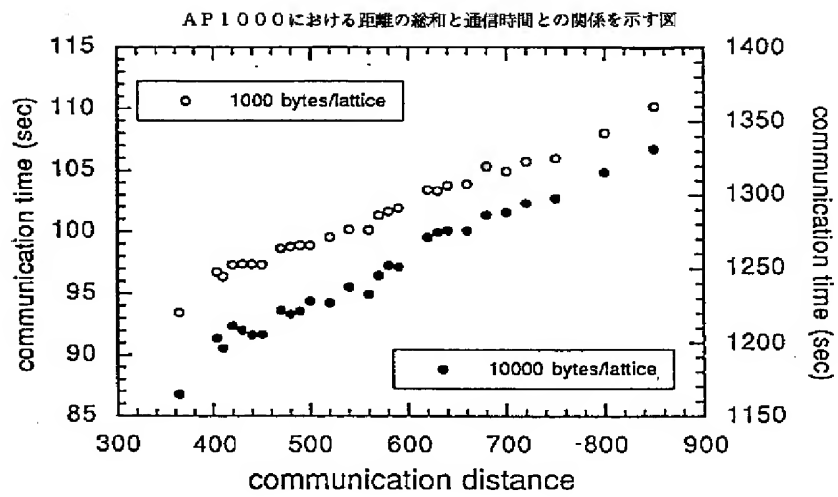


【図14】

AP1000における距離と時間との関係を示す図

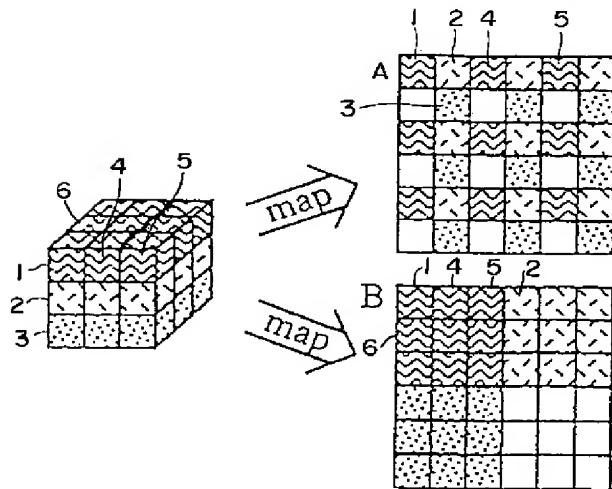


【図15】



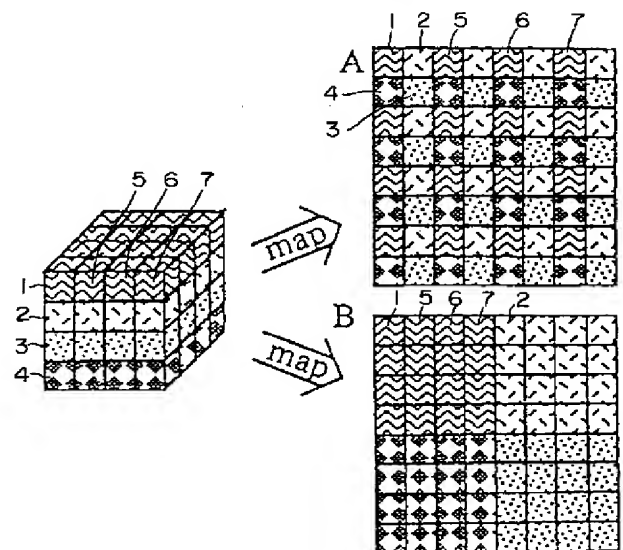
【図16】

3°を8°にマッピングした場合を示す図



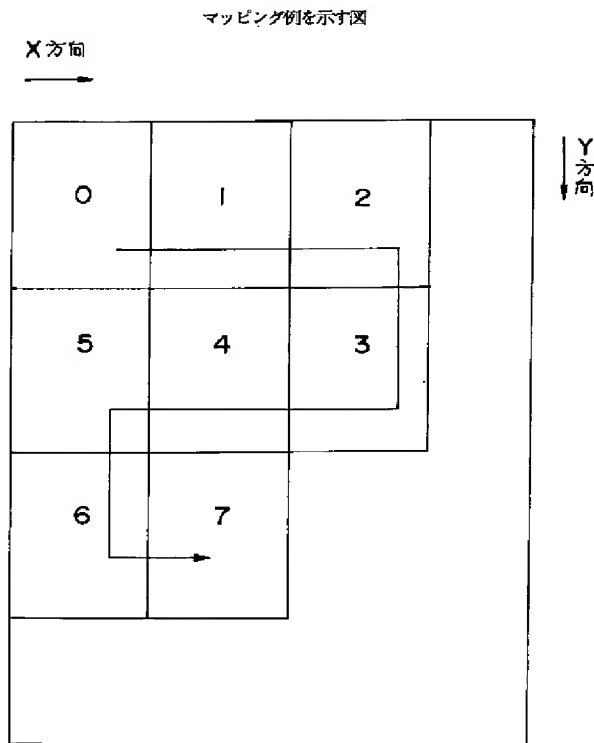
【図17】

4°を8°にマッピングした場合を示す図

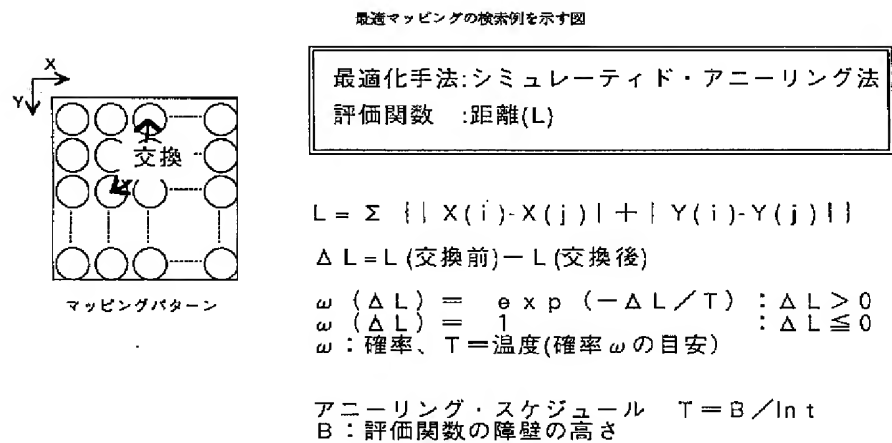




【図19】

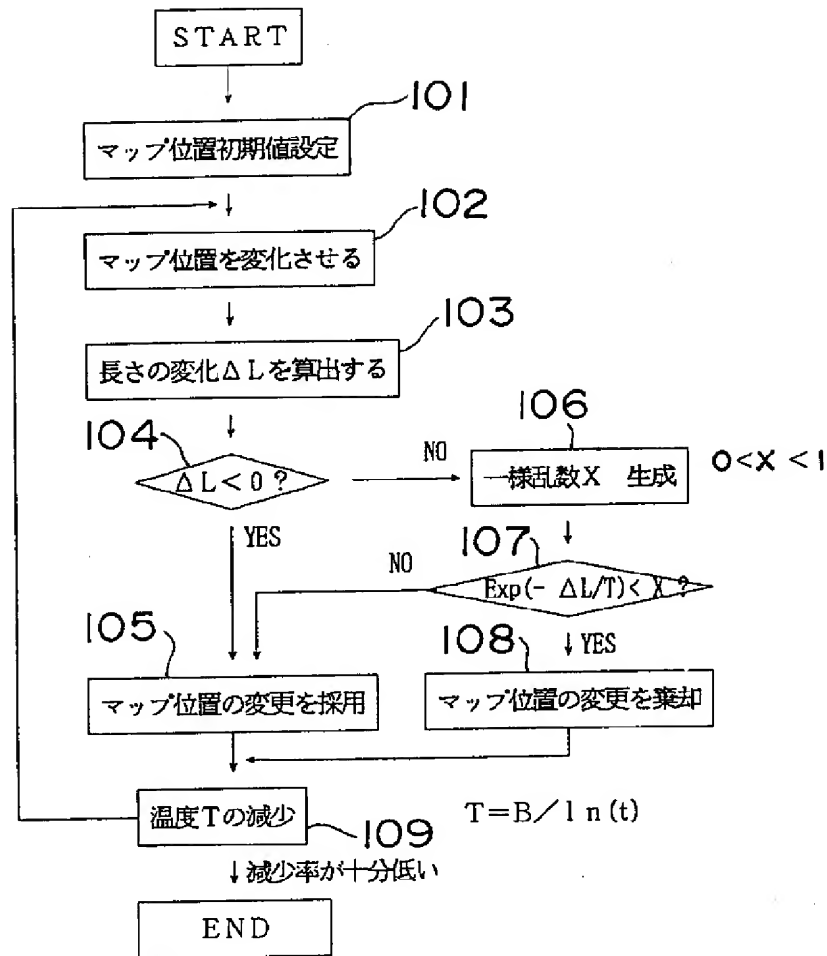


【図20】



【図21】

シミュレーテド・アニーリング法のアルゴリズムを示す図

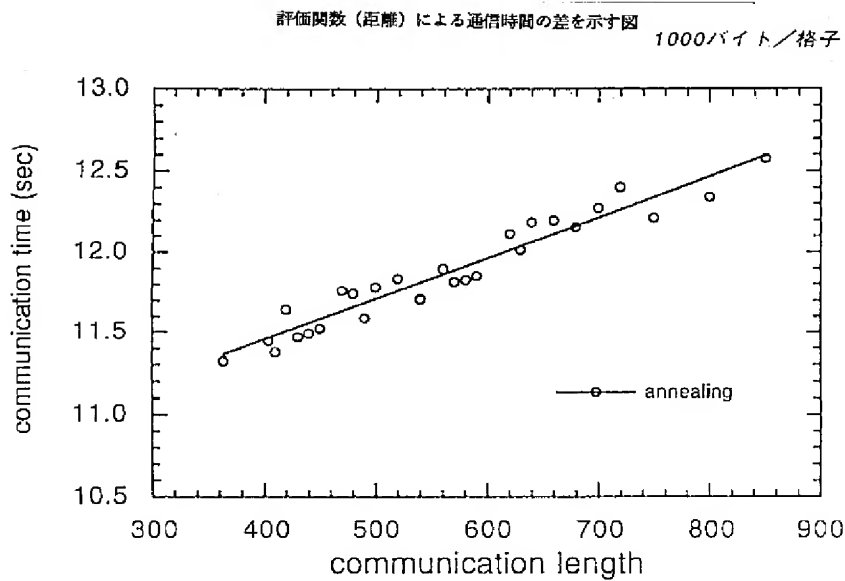


【図29】

マッピングの手法による評価関数と実測の通信時間との関係を示した図

手法	マッピング パターン	系	距離の総和 L
頭で考えた マッピング	A	$3^3 \rightarrow 8^2$	180
		$4^3 \rightarrow 8^2$	320
	B	$3^3 \rightarrow 8^2$	180
		$4^3 \rightarrow 8^2$	448
アニーリング	C	$3^3 \rightarrow 8^2$	154
	D	$4^3 \rightarrow 8^2$	320

【図22】



【図23】

$3^3$ を $8^2$ にアニーリング法でマッピングした場合の具体的結果を示す図

```

1 mapfile is a27.154
2 # 3 dimensional mapping
3 # 10000 bytes per a lattice
4 # 100 steps
5 # send-direction 6
6 # LOCELX= 3
7 # LOCELY= 3
8 # LOCELZ= 3
9 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
10 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
11 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
12 (-,-,-) (-,-,-) (-,-,-) (2,2,1) (1,2,1) (-,-,-) (-,-,-) (-,-,-)
13 (0,2,0) (2,2,0) (1,2,0) (2,1,1) (2,2,2) (1,2,2) (0,2,2) (0,2,1)
14 (0,1,0) (2,1,0) (1,1,0) (1,1,1) (2,1,2) (1,1,2) (0,1,2) (0,1,1)
15 (0,0,0) (2,0,0) (1,0,0) (1,0,1) (2,0,2) (1,0,2) (0,0,2) (0,0,1)
16 (-,-,-) (-,-,-) (-,-,-) (2,0,1) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
17 # creating task on 27 PE
18 # sending parameter
19 # sending logical position to each cell
20 # sending logical mapping
21 # waiting stable
22 *** HOST_EXIT(0). ***
23 -----
24 Max: 2.037274 Min: 2.033902 Ave: 2.035016 → 実測時間

```

マッピングパターン

【図24】

$3^3$ を $8^2$ にスキップ法でマッピングした場合の具体的結果を示す図

```

26 mapfile is map.skip.27
27 # 3 dimensional mapping
28 # 10000 bytes per a lattice
29 # 100 steps
30 # send-direction 6
31 # LOCELX= 3
32 # LOCELY= 3
33 # LOCELZ= 3
34 (0,0,0) (0,0,1) (0,1,0) (0,1,1) (0,2,0) (0,2,1) (-,-,-) (-,-,-)
35 (-,-,-) (0,0,2) (-,-,-) (0,1,2) (-,-,-) (0,2,2) (-,-,-) (-,-,-)
36 (1,0,0) (1,0,1) (1,1,0) (1,1,1) (1,2,0) (1,2,1) (-,-,-) (-,-,-)
37 (-,-,-) (1,0,2) (-,-,-) (1,1,2) (-,-,-) (1,2,2) (-,-,-) (-,-,-)
38 (2,0,0) (2,0,1) (2,1,0) (2,1,1) (2,2,0) (2,2,1) (-,-,-) (-,-,-)
39 (-,-,-) (2,0,2) (-,-,-) (2,1,2) (-,-,-) (2,2,2) (-,-,-) (-,-,-)
40 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
41 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
42 # creating task on 27 PE
43 # sending parameter
44 # sending logical position to each cell
45 # sending logical mapping
46 # waiting stable
47 *** HOST_EXIT(0) ***
48 -----
49 Max: 1.805308 Min: 1.802989 Ave: 1.803961  → 実行時間

```

マッピングパターン

【図25】

$3^3$ を $8^2$ にタイル法でマッピングした場合の具体的結果を示す図

```

51 mapfile is map.tile.27
52 # 3 dimensional mapping
53 # 10000 bytes per a lattice
54 # 100 steps
55 # send-direction 6
56 # LOCELX= 3
57 # LOCELY= 3
58 # LOCELZ= 3
59 (0,0,0) (0,1,0) (0,2,0) (0,0,1) (0,1,1) (0,2,1) (-,-,-) (-,-,-)
60 (1,0,0) (1,1,0) (1,2,0) (1,0,1) (1,1,1) (1,2,1) (-,-,-) (-,-,-)
61 (2,0,0) (2,1,0) (2,2,0) (2,0,1) (2,1,1) (2,2,1) (-,-,-) (-,-,-)
62 (-,-,-) (-,-,-) (-,-,-) (0,0,2) (0,1,2) (0,2,2) (-,-,-) (-,-,-)
63 (-,-,-) (-,-,-) (-,-,-) (1,0,2) (1,1,2) (1,2,2) (-,-,-) (-,-,-)
64 (-,-,-) (-,-,-) (-,-,-) (2,0,2) (2,1,2) (2,2,2) (-,-,-) (-,-,-)
65 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
66 (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-) (-,-,-)
67 # creating task on 27 PE
68 # sending parameter
69 # sending logical position to each cell
70 # sending logical mapping
71 # waiting stable
72 *** HOST_EXIT(0) ***
73 -----
74 Max: 1.752943 Min: 1.748959 Ave: 1.750421  → 実行時間
75 -----

```

マッピングパターン

【図26】

4<sup>3</sup>を8<sup>2</sup>にアニーリング法でマッピングした場合の具体的結果を示す図

```

1  mapfile is a64.320
2  # 3 dimensional mapping
3  # 10000 bytes per a lattice
4  # 100 steps
5  # send-direction 6
6  # LOCELX= 4
7  # LOCELY= 4
8  # LOCELZ= 4
9  (3,1,0) (2,2,0) (3,2,0) (3,2,3) (2,2,3) (3,1,3) (2,1,3) (2,1,0)
10 (3,1,1) (2,2,1) (3,2,1) (3,2,2) (2,2,2) (3,1,2) (2,1,2) (2,1,1)
11 (0,1,1) (1,2,1) (0,2,1) (0,2,2) (1,2,2) (0,1,2) (1,1,2) (1,1,1)
12 (3,0,1) (2,3,1) (3,3,1) (3,3,2) (2,3,2) (3,0,2) (2,0,2) (2,0,1)
13 (0,0,1) (1,3,1) (0,3,1) (0,3,2) (1,3,2) (0,0,2) (1,0,2) (1,0,1)
14 (0,0,0) (1,3,0) (0,3,0) (0,3,3) (1,3,3) (0,0,3) (1,0,3) (1,0,0)
15 (3,0,0) (2,3,0) (3,3,0) (3,3,3) (2,3,3) (3,0,3) (2,0,3) (2,0,0)
16 (0,1,0) (1,2,0) (0,2,0) (0,2,3) (1,2,3) (0,1,3) (1,1,3) (1,1,0)
17 # creating task on 64 PE
18 # sending parameter
19 # sending logical position to each cell
20 # sending logical mapping
21 # waiting stable
22 *** HOST_EXIT(0) ***
23 -----
24 Max: 1.819263 Min: 1.817908 Ave: 1.818579

```

マッピングパターン

【図27】

4<sup>3</sup>を8<sup>2</sup>にスキップ法でマッピングした場合の具体的結果を示す図

```

26 mapfile is map.skip.64
27 # 3 dimensional mapping
28 # 10000 bytes per a lattice
29 # 100 steps
30 # send-direction 6
31 # LOCELX= 4
32 # LOCELY= 4
33 # LOCELZ= 4
34 (0,0,0) (0,0,1) (0,1,0) (0,1,1) (0,2,0) (0,2,1) (0,3,0) (0,3,1)
35 (0,0,3) (0,0,2) (0,1,3) (0,1,2) (0,2,3) (0,2,2) (0,3,3) (0,3,2)
36 (1,0,0) (1,0,1) (1,1,0) (1,1,1) (1,2,0) (1,2,1) (1,3,0) (1,3,1)
37 (1,0,3) (1,0,2) (1,1,3) (1,1,2) (1,2,3) (1,2,2) (1,3,3) (1,3,2)
38 (2,0,0) (2,0,1) (2,1,0) (2,1,1) (2,2,0) (2,2,1) (2,3,0) (2,3,1)
39 (2,0,3) (2,0,2) (2,1,3) (2,1,2) (2,2,3) (2,2,2) (2,3,3) (2,3,2)
40 (3,0,0) (3,0,1) (3,1,0) (3,1,1) (3,2,0) (3,2,1) (3,3,0) (3,3,1)
41 (3,0,3) (3,0,2) (3,1,3) (3,1,2) (3,2,3) (3,2,2) (3,3,3) (3,3,2)
42 # creating task on 64 PE
43 # sending parameter
44 # sending logical position to each cell
45 # sending logical mapping
46 # waiting stable
47 *** HOST_EXIT(0) ***
48 -----
49 Max: 1.679813 Min: 1.679192 Ave: 1.679333
50 -----

```

【図28】

$4^3$ を $8^2$ にタイル法でマッピングした場合の具体的結果を示す図

```

52 # 3 dimensional mapping
53 # 10000 bytes per a lattice
54 # 100 steps
55 # send-direction 6
56 # LOCELX= 4
57 # LOCELY= 4
58 # LOCELZ= 4
59 (0,0,0) (0,1,0) (0,2,0) (0,3,0) (0,0,1) (0,1,1) (0,2,1) (0,3,1)
60 (1,0,0) (1,1,0) (1,2,0) (1,3,0) (1,0,1) (1,1,1) (1,2,1) (1,3,1)
61 (2,0,0) (2,1,0) (2,2,0) (2,3,0) (2,0,1) (2,1,1) (2,2,1) (2,3,1)
62 (3,0,0) (3,1,0) (3,2,0) (3,3,0) (3,0,1) (3,1,1) (3,2,1) (3,3,1)
63 (0,0,3) (0,1,3) (0,2,3) (0,3,3) (0,0,2) (0,1,2) (0,2,2) (0,3,2)
64 (1,0,3) (1,1,3) (1,2,3) (1,3,3) (1,0,2) (1,1,2) (1,2,2) (1,3,2)
65 (2,0,3) (2,1,3) (2,2,3) (2,3,3) (2,0,2) (2,1,2) (2,2,2) (2,3,2)
66 (3,0,3) (3,1,3) (3,2,3) (3,3,3) (3,0,2) (3,1,2) (3,2,2) (3,3,2)
67 # creating task on 64 PE
68 # sending parameter
69 # sending logical position to each cell
70 # sending logical mapping
71 # waiting stable
72 *** HOST_EXIT(0) ***
73 -----
74 Max: 1.887382 Min: 1.884576 Ave: 1.885689
75 -----

```

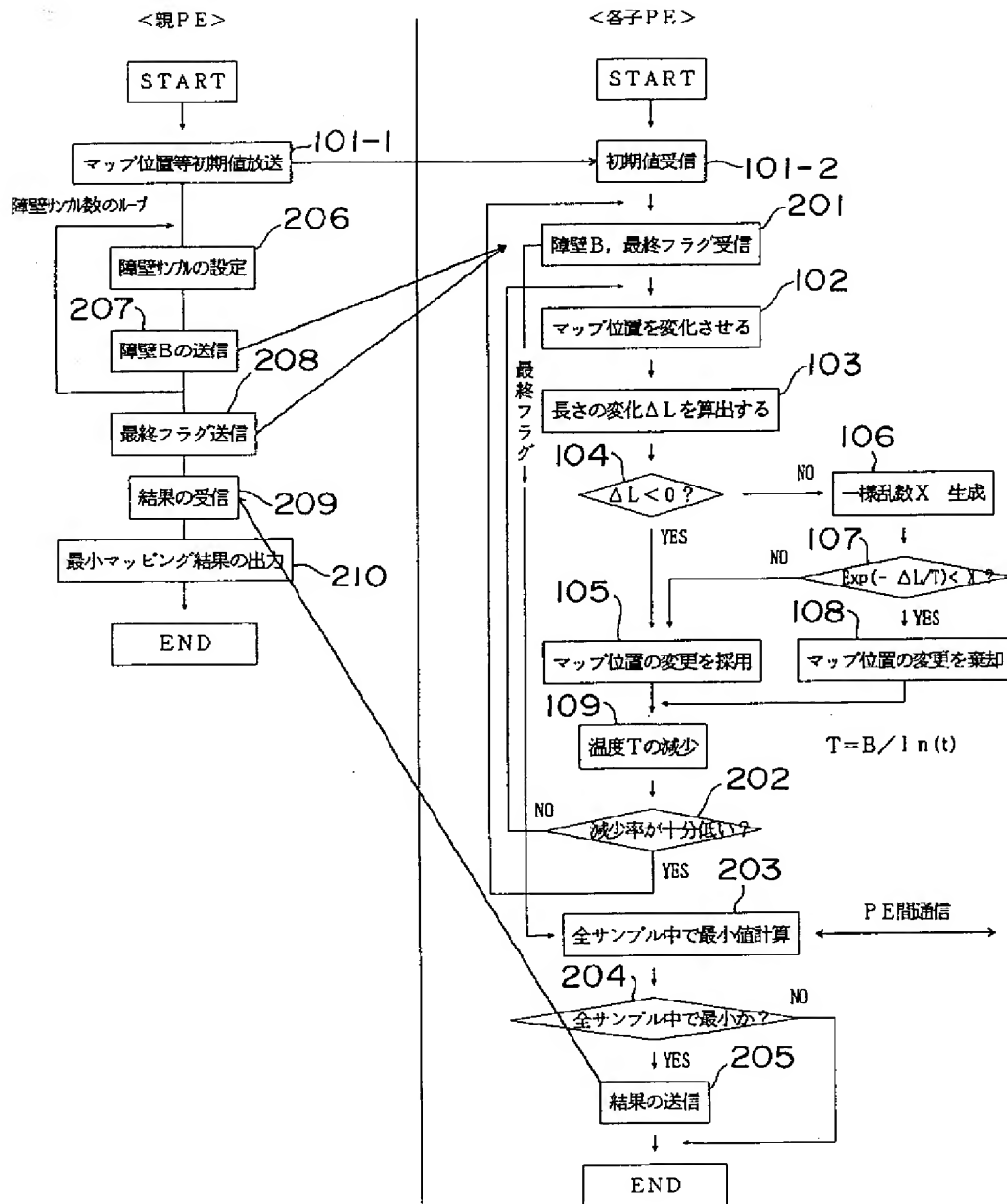
【図30】

最適マッピング情報データベースのファイル形式を示した図

1. 取得情報種別 (入力キー)
  - 1: ユニット分割次元数
  - 2: マッピングテーブル
2. 計算モデル次元数: M (入力キー)
  - 1~10
3. データ参照関係: R (入力キー)
  - 1: 最隣接 PE
  - 2: 近接 PE (斜め横を含む)
  - 3: 全 PE
  - 4以上: ユーザ指定  
自 PE からの間接アドレスで参照のある PE を指定する。(0, -1), (1, -1), ...
4. PE 台数: N (入力キー)
  - 1~1024
5. ユニット単位の転送データ量: D (入力キー)
  - 1~4: 1~10000 (バイト)
  - 5以上: ユーザ指定
6. ユニット分割次元数: UM (出力1)
  - 1~M-1
7. マッピングテーブル (出力2)
 

論理 PE 番号    ブロックユニット番号

【図31】



フロントページの続き

(72)発明者 奥田 基  
 神奈川県川崎市中原区上小田中1015番地  
 富士通株式会社内